НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО

Факультет інформатики та обчислювальної техніки
(назва факультету, інституту)

Кафедра автоматизованих систем обробки інформації і управління
(назва кафедри)

"На правах рукопису"
УДК 004.023: 519.254

«До захисту допущено»
В.о.завідувача кафедри

_____ О.А.Павлов
(підпис) (ініціали, прізвище)

“____” _____ 20*19* р.

МАГІСТЕРСЬКА ДИСЕРТАЦІЯ

на здобуття ступеня магістра

за спеціальністю *126 Інформаційні системи та технології*
(код та назва спеціальності)

ОПП *Інформаційні управляючі системи та технології*
(код та назва спеціалізації)

на тему: *Інтелектуальна система підтримки вибору житла для покупців та орендаторів*

Виконав: студент *VI* курсу групи ІС-81мп
(шифр групи)

Меліков Євгеній Олексійович _____
(прізвище, ім’я, по батькові) (підпис)

Науковий керівник *Доцент, к.т.н., доцент Тєлишева Т. О.* _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант *Проф., д.т.н., проф. Томашевський В.М.* _____
(науковий ступінь, вчене звання, прізвище, ініціали) (підпис)

Рецензент *доцент, к.т.н., доцент Ткач М.М.* _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2019

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ім. ІГОРЯ СІКОРСЬКОГО»

Факультет *інформатики та обчислювальної техніки*
(повна назва)

Кафедра *автоматизованих систем обробки інформації та управління*
(повна назва)

Рівень вищої освіти *другий (магістерський) за освітньо-професійною програмою*

Спеціальність *126 Інформаційні системи та технології*
(код і назва)

ОПП *Інформаційні управляючі системи та технології*
(код і назва)

ЗАТВЕРДЖУЮ
В.о.завідувача кафедри
_____ О.А.Павлов
(підпис) (ініціали, прізвище)
«__» _____20*19* р.

ЗАВДАННЯ

на магістерську дисертацію студенту

Мелікову Євгенію Олексійовичу
(прізвище, ім'я, по батькові)

1. Тема дисертації *Інтелектуальна система підтримки вибору житла для покупців та орендаторів*

науковий керівник дисертації *Тєлишева Тамара Олексіївна, доцент, к.т.н.*
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від "*28*" *жовтня* 20*19* р. № 3770-с

2. Строк подання студентом дисертації "*2*" *12* 20*19* р.

3. Об'єкт дослідження *процес пошуку житла для оренди*

4. Перелік завдань, які потрібно розробити
провести аналіз існуючих систем оцінки пропозиції житла;
поставити вимоги до системи;
розробити алгоритм опрацювання даних та фільтрів;
розробити зручний інтерфейс користувача.

5. Орієнтовний перелік ілюстративного матеріалу *Екранні форми, діаграма послідовності, діаграма станів, діаграма прецедентів, архітектура системи, концептуальна схема бази даних, математична модель.*

6. Орієнтовний перелік публікацій *стаття та тези доповіді на науковій конференції*

7. Консультанти розділів дисертації

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

8. Дата видачі завдання "*2*" *вересня* 20*19* р.

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Строк виконання етапів магістерської дисертації	Примітка
1	*Систематизація результатів огляду літератури*	*09.09.2019*	
2	*Порівняльний аналіз існуючих методів розв'язання задачі*	*16.09.2019*	
3	*Постановка та формалізація математичної моделі задачі*	*23.09.2019*	
4	*Модифікація існуючих методів розв'язання задачі*	*30.09.2019*	
5	*Розробка інформаційного та програмного забезпечення*	*21.10.2019*	
7	*Проведення експериментальних досліджень розроблених алгоритмів*	*28.10.2019*	
8	*Оформлення документації*	*11.11.2019*	
9	*Подання роботи на попередній захист*	*20.11*	
10	*Подання роботи на основний захист*	*02.12*	

Студент _____ *Є. О. Меліков*
(підпис) (ініціали, прізвище)

Науковий керівник _____ *Т. О. Тєлишева*
(підпис) (ініціали, прізвище)

РЕФЕРАТ

Магістерська дисертація: 90 с., 13 рис., 26 табл., 11 джерел.

Актуальність. На сьогодні велика кількість людей перебуває в пошуку житла. Житло має бути не занадто дорогим, зручно розташованим в мікрорайоні з достатньо розвиненою інфраструктурою та зручним транспортом. Для іншого житло має бути за межею міста та поряд з лісом і т. д. Кожен шукає житло по своїм критеріям.

Зазвичай, для оренди квартири використовують сайти з фільтрами пропозицій, за допомогою яких значно спрощується пошук кращого рішення. Але такі фільтра мають бути надскладними щоб задовольнити всі бажання користувачів. Більш того, користувачу важко оцінити чи вартість житла відповідає нормі чи є завищеною.

При виборі житла витрачається велика кількість часу та зусиль, хоча значну частину цього процесу можна автоматизувати. Система має представити вибір з пропозицій що найбільше підходять під бажання клієнта.

Проблемою є неточність та не гнучкість алгоритмів для пошуку житла; час, що витрачається користувачем, на оцінку всіх рішень та зіставлення їх з очікуваннями користувача.

Зв'язок роботи з науковими програмами, планами, темами. Робота виконувалась на кафедрі автоматизованих систем обробки інформації та управління Національного технічного університету України «Київський політехнічний інститут ім. Ігоря Сікорського» в рамках теми «Інтелектуальна система підтримки вибору житла для покупців та орендаторів».

Мета дослідження – оптимізація процесу пошуку найкращої пропозиції з великого об’єму даних за рахунок використання інтелектуальних алгоритмів з одного боку та спрощення процесу надання житла в з іншого.

Для досягнення мети необхідно виконати наступні **завдання**:

- оптимізація вибору житла для оренди;

- оптимізація формування пропозиції для оренди;
- спрощення процесу укладення угоди про оренду;
- спрощення процесу оплати оренди.

Об’єкт дослідження – процес пошуку житла для оренди.

Предмет дослідження – методи кластеризації та класифікації використовуючи регресійні моделі як оціночні функції для класифікації.

Наукова новизна отриманих результатів

Розроблено підходи та методи вирішення поставленої задачі із використанням методів кластерізації та класифікації. Використання створених регресійних моделей дозволяє істотно підвищити якість відносної оцінки однорідних даних.

КЛАСИФІКАЦІЯ, КЛАСТЕРІЗАЦІЯ, РЕГРЕСІЙНИЙ АНАЛІЗ, С-СЕРЕДНІХ, К-СЕРЕДНІХ, РЕГРЕСІЙНА МОДЕЛЬ

ABSTRACT

Master's thesis: 90 pages, 13 figures, 26 tables, 11 sources.

Actuality. Nowadays, a large number of people are in search of housing. The housing should be not too expensive, conveniently located in a neighborhood with a well-developed infrastructure and convenient transportation. For someone other, housing should be outside the city and near the forest, etc. Everyone is looking for housing according to his or her own criteria.

Typically, rental properties use sites with offer filters that make it much easier to find the best solution. However, such filters must be sophisticated in order to satisfy all user desires. Moreover, it is difficult for the user to assess whether the cost of housing is in line with the standard or overpriced.

Choosing a home takes a lot of time and effort, though much of this process can be automated. The system should provide a selection of the proposals most suited to the client's wishes.

The problem is the inaccuracy and non-flexibility of housing search algorithms; time spent by the user to evaluate all the solutions and compare them to the user's expectations.

Relationship with scientific programs, plans, topics. The work was performed at the department of automated information processing and management systems of the national technical university of Ukraine « Igor Sikorsky Kyiv Polytechnic Institute" within the topic "Intelligent Support System for Buyers and Tenants Making Housing Choices".

The purpose of the study is to optimize the process of finding the best deal on a large amount of data by using intelligent algorithms on the one hand and simplifying the process of renting housing through a user-friendly interface on the other.

To achieve this goal, you must complete the following **tasks**:

- optimization of housing selection for rent;
- optimization of the formation of a rental offer;
- simplifying the lease process;
- simplifying the lease payment process.

Object of study - the process of finding housing for rent.

The subject of the study is clustering methods and mining date using regression models as estimation functions for clustering.

Scientific novelty of the obtained results

Approaches and methods of solution of the given problem with use of clustering methods and classification are developed. Using created regression models can significantly improve the quality of relative estimation of homogeneous data.

CLASSIFICATION, CLUSTERING, REGRESSION ANALYSIS, C-MEDIUM, K-MEDIUM, REGRESSION MODEL

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

СУБД – Система Управління Базами Даних

БД – База Даних

МБ – Мега Байт

ПЗ – Програмне Забезпечення

API – Application Programming Interface

SDK – Software Development Kit

MVC – Model View Controller Pattern

JSX – JavaScript XML

CSS – Cascading Style Sheets

SASS (SCSS) – syntactically Awesome Stylesheets

JSON – JavaScript Object Notation

ORM – Object-Relational Mapping

DOM – Document Object Model

ВСТУП

На сьогодні велика кількість людей перебуває в пошуку житла. Вибір житла є непростою задачею для людей, так як кількість критеріїв може бути великою, а оцінити усі наявні пропозиції не завжди можливо. Критеріїв обрання житла є безліч і вагомій внесок у спрощення внесли фільтри, які можуть допомогти обрати серед усіх наявних пропозицій невелику вибірку, серед якої і відбувається вибір.

Але такі фільтри мають бути надскладними щоб задовольнити всі бажання користувачів. Більш того, користувачу важко оцінити чи вартість житла відповідає нормі чи є завищеною. Проте така система також не завжди є успішною,

При виборі житла витрачається велика кількість часу та зусиль, хоча значну частину цього процесу можна автоматизувати та зробити інтуїтивно зрозумілою. Система має представити вибір з пропозицій що найбільше підходять під бажання клієнта.

Проблемою є неточність та не гнучкість алгоритмів для пошуку житла; час, що витрачається користувачем, на оцінку всіх рішень та зіставлення їх з очікуваннями користувача.

Ще одним вагомим фактором є те, що продавці та орендодавці інколи не можуть правильно розібратись із заповненням своєї пропозиції. Звідси виникає проблема, відсутність правильного опису вводить в оману користувача і орендодавця, що призводить до втрати часу та довіри. Саме тому є актуальним впровадження системи пошуку житла на основі інтелектуального алгоритму, що сприятиме покращенню пошуку житла шляхом фільтрів.

Метою дослідження є покращення процесу пошуку найкращої пропозиції житла з великого об’єму даних за рахунок використання інтелектуальних алгоритмів та спрощення процесу надання житла в оренду за допомогою інтерфейсу.

Для досягнення поставленої мети було визначено наступні завдання:

- провести аналіз існуючих систем оцінки пропозиції житла;

- поставити вимоги до системи;
- розробити алгоритм опрацювання даних та фільтрів;
- розробити зручний інтерфейс користувача.

Об’єкт дослідження – процес пошуку житла для оренди.

Предмет дослідження – методи кластеризації та класифікації використовуючи регресійні моделі як оціночні функції для класифікації.

Методологічною основою для дослідження стали методи: порівняльні методи, опитування експертів, методи синтезу, аналізу та спостереження.

1 ПРОЕКТНІ РІШЕННЯ З РОЗРОБКИ СИСТЕМИ ПІДТРИМКИ ВИБОРУ ЖИТЛА ДЛЯ ПОКУПЦІВ ТА ОРЕНДАТОРІВ

1.1 Опис бізнес – процесів

1.1.1 Опис процесу діяльності

Основна задача системи – інтелектуальний пошук з великого об’єму даних пропозицій за заданими критеріями. Користувач, що потрапляє на сайт системи, може ввести параметри, що цікавлять його в системі, та отримати список пропозицій що найкраще підходять під його запит.

Ще однією задачею системи є багатостороння оцінка кожної пропозиції з метою визначення її якості відносно ціни. Ця інформація разом з побажаннями користувача використовується для побудови списку пропозицій для кожного окремого користувача.

Користувачі можуть створити обліковий запис в системі надавши інформацію про себе. Зареєстровані користувачі можуть створювати нові пропозиції для оренди житла в системі.

При створенні пропозиції користувачу розраховується приблизна оцінка якості його пропозиції відносно ціни та надається рекомендована ціна для його пропозиції.

Користувачі можуть здійснити оренду житла з пропозиції якщо обидві сторони є згодними з наданими умовами та ціною. У такому випадку користувач-орендар та користувач-орендодавець мають надати згоду на здійснення оренди. Після цього орендар має сплатити домовлену суму оренди через систему. У випадку не сплати – відбувається відміна згоди.

У випадку прострочення регулярного платежу оренди користувачу надається штраф та відмічається у обліковому записі користувача. Ця інформація буде доступною наступним орендодавцям з якими може бути здійснення оренди в майбутньому.

Користувач-орендодавець може прибрати свою пропозицію з активного списку, але залишити її у своєму архіві. В такому випадку користувачі-орендарі не зможуть знайти дану пропозицію та здійснити її оренду.

1.1.2 Предметна область

Предметна область – процес пошуку житла для оренди.

Основним об’єктом в системі - є пропозиція. Кожна пропозиція будується за визначеним шаблоном. В шаблоні є ряд необхідних параметрів як площа, кількість кімнат, тип ремонту, тощо. Також є ряд не обов’язкових параметрів які орендодавець може вказувати за бажанням.

Всі процеси виконуються з даними пропозиції та процесами її оренди.

Задля оренди користувачі мають погодитись з правилами використання системи. Це є обов’язковим для завершення створення облікового запису.

1.1.3 Актори і функції

Система має такі групи акторів (рисунок 1.1):

- орендодавці;
- орендарі;
- незареєстровані користувачі.

Всі функції та взаємодії в системі відбуваються між вказаними групами акторів.

Орендодавці мають такі функції:

- створення пропозиції для оренди;
- редагування пропозиції для оренди;
- сховати\показати пропозицію для орендарів;
- створення облікового запису;
- редагування облікового запису.

Рисунок 1.1 – Діаграма прецедентів

Створення пропозиції відбувається за визначеною системою процедурою та складається з двох етапів. Перший – наведення інформації про пропозицію за шаблоном. Другий – погодження з умовами системи та погодження на обробку та публікацію інформації про пропозицію в системі.

Редагування пропозиції – це можливість орендодавця змінити поля шаблону пропозиції оренди.

Функція орендодавця сховати або показати пропозицію для орендарів – це здатність тимчасово вилучити пропозицію з системи інтелектуального вибору житла. Проте пропозиція буде використовуватись в алгоритмах системи.

Редагування облікового запису – це здатність користувача змінити персональні дані, контакти, фотографію, тощо.

Орендар має наступні функції:

– перегляд пропозиції;
– фільтрація пропозицій;

- вибірка пропозицій;
- редагування облікового запису;
- здійснення оренди;
- оплата діючої оренди.

Відповідно можливість перегляду пропозиції – це здатність до отримування інформації про всі пропозиції в системі.

Функція фільтрації пропозицій – це можливість відсіювання пропозицій за заданими критеріями для звуження вибірки пропозицій до прийнятних варіантів та меншого об’єму даних.

Редагування облікового запису є функцією ідентичною до аналогічної функції орендодавця.

Здійснення оренди - це функція орендаря, яка полягаю в укладенні угоди з системою на виконання певних домовленостей з сторони орендаря та надання житла з іншої сторони.

Оплата діючої оренди є функцією, що виконується під час дії угоди про оренду та полягає в сплаті певної суми коштів в певний час, згідно угоди.

Незареєстрований користувач має наступні функції:

- перегляд пропозицій;
- фільтрація пропозицій;
- вибірка пропозицій;
- створення облікового запису.

Перегляд пропозицій, фільтрація пропозицій, інтелектуальна вибірка пропозицій є ідентичними до відповідних функцій орендаря.

Створення облікового запису – це функція перетворення анонімного користувача на орендаря або орендодавця та полягає в наданні інформації користувача про себе.

На діаграмі прецедентів відображено функціональні вимоги до системи, але слід зазначити, що дана діаграма відображає систему поверхнево і не відображає усіх наявних класів всередині системи.

1.1.4 Огляд основних бізнес процесів

Одним з основних завдань системи для користувачів є підписання договору (рисунок 1.2). Даний бізнес-процес вимагає чіткого виконання, критично важливим фактором є швидкість його виконання - будь-які зволікання при визначенні орендарів позначаються в простої площ, а значить, у втраті прибутку. Крім того, в ході даного процесу здійснюється робота з документами - без належного контролю ця задача нерідко виливається в ряд проблем: документи оформляються невірно, нерідко з використання неправильних або вже неактуальних даних, паперу щезають іноді незрозуміло, хто повинен виконувати ту чи іншу задачу.

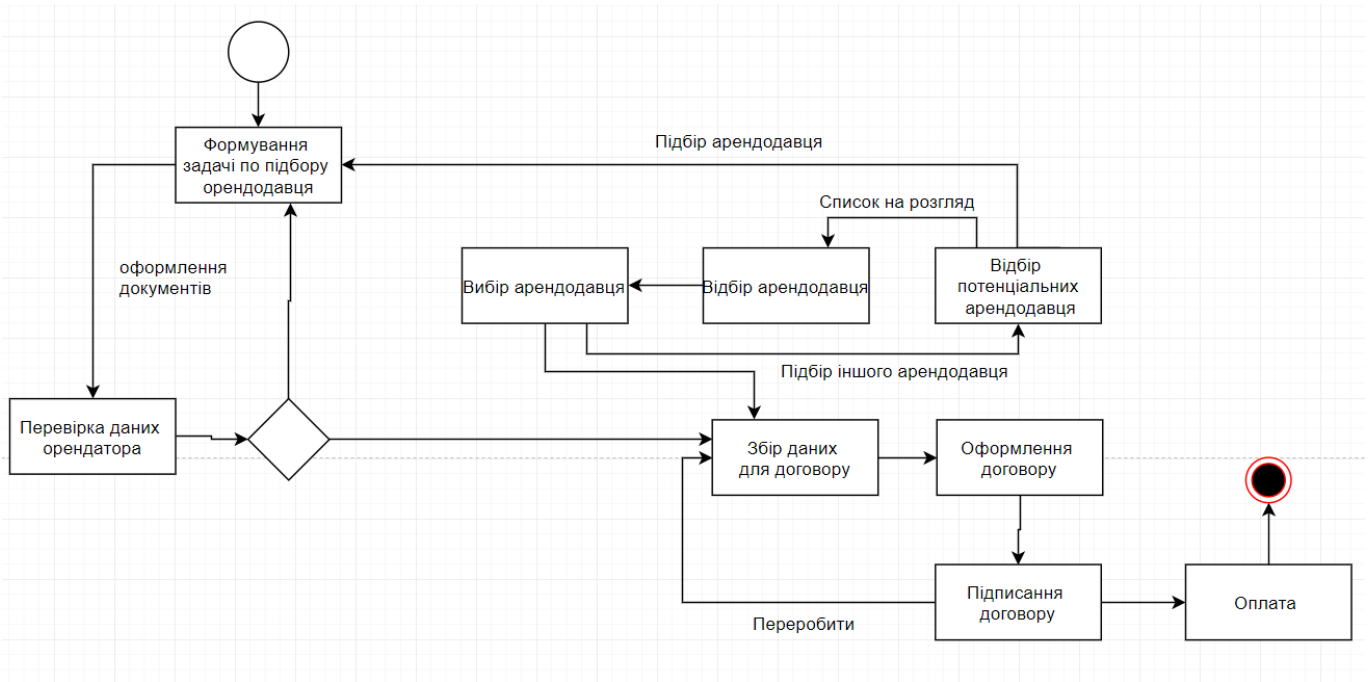


Рисунок 1.2 – Діаграма станів

Виконання бізнес-процесу починається з формування завдання укладення договору оренди. При цьому користувач відразу отримає у відповідних полях форми завдання детальну актуальну інформацію про вибраний об'єкт оренди: найменування, поверх, розташування орендованого приміщення, ім'я попереднього орендаря. Різна інформація, яка відображається на сторінці користувача завдання може бути

завантажена з різних джерел і систем. При цьому користувач не відволікається на технічні моменти функціонування системи - йому все одно, звідки були отримані ті чи інші дані - він бачить все в одному місці; від нього вимагається лише ознайомитися з деталями завдання, виконати зазначені доручення, а потім доповісти системі про виконання завдання.

Наступний бізнес-процес – надання вибірки варіантів користувачу на його запит (рисунок 1.3). Він починається з надання пошуковій системі запиту на пошук та задання клієнтом відповідних фільтрів на пошук житла. Від пошукової системи надається запит в базу даних, де серед усіх варіантів обирається ті які відповідають критеріям користувача. Від бази даних видаються варіанти і з них формується та виводиться відсортована вибірка пропозицій.

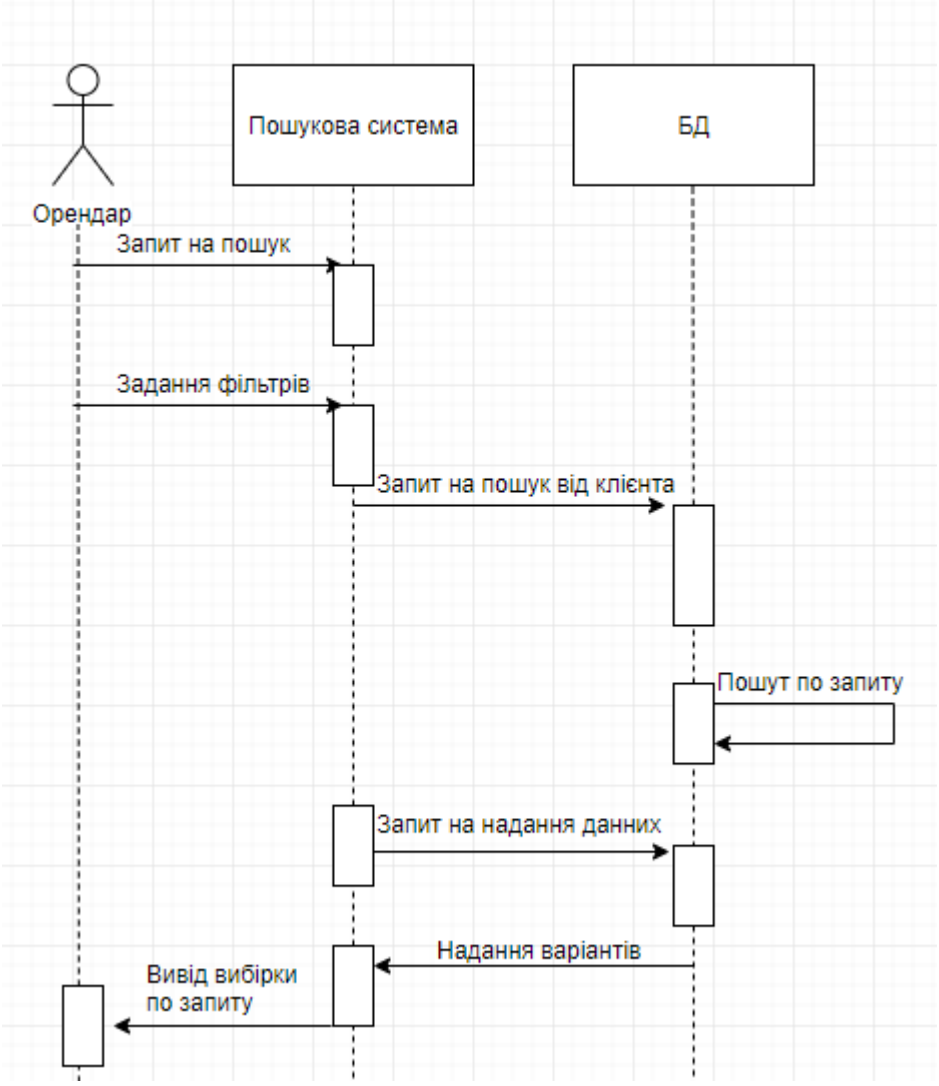


Рисунок 1.3 – Бізнес-процес – пошук пропозицій

1.2 Огляд існуючих систем

Для оцінки існуючих систем було розглянуто декілька рішень:

1) Система «T.H.E. Capital»

Система надає пропозиції для продажу та оренди. Користувач може звузити список пропозицій за наступними критеріями: ціна, площа, район, кількість кімнат, назва житлового комплексу, тип ремонту, наявність меблів, поверх.

Вибір за житловим комплексом є актуальним, оскільки житловий комплекс надає достатньо інформації про інфраструктуру та приблизний рівень якості житла. Проте фільтр за районом не функціональний, оскільки райони занадто великі та не несуть інформації про пропозицію як мікрорайони та житлові комплекси.

Також дана система не має можливості шукати пропозиції за відстані до певних станцій метро. Можливості пошуку за елементами інфраструктури також немає (Наприклад: спортзал з басейном).

2) Система «Park Lane»

Система має розширений фільтр з можливістю вибору ціни, площі, площі кухні, адреси, станції метро, кількість кімнат, поверх, ремонт та меблі. Це майже достатній набір фільтрів, щоб однозначно вибрати житло. Проте дана система має декілька суттєвих недоліків.

Зазвичай, користувач не знає однозначно яка станція метро йому підходить і точна адреса також не цікавить. Замість цього може бути адреса місця роботи або інші місця, що мають бути легко досяжними.

Пошуку за інфраструктурою мікрорайону також немає.

3) Система «OLX»

Одна з найпоширеніших торгових площадок, проте має занадто простий фільтр: площа, ціна, поверх та район.

За такими критеріями користувачу буде необхідно самостійно продумати можливі маршрути (для роботи, університету, тощо) для кожної з пропозицій що задовільнять фільтр. А при потребах в певних елементах інфраструктури мікрорайону буде ще складніше. На пошуки кращої пропозиції в даній системі користувач буде вимушений передивитись тисячі варіантів.

4) Система «Лун»

Система має спрощений фільтр, аналогічний до системи «T.H.E. Capital», проте всі відфільтровані пропозиції презентовані на мапі, що значно спрощує вибір, оскільки даний варіант є більш наглядним ніж адреса.

На мапі показані деякі елементи інфраструктури, що можуть зацікавити користувача. Проте дуже мало, немає магазинів та супермаркетів, немає позначень парків, учбових закладів, дитячих майданчиків, спортивних майданчиків, тощо.

З оцінки існуючих систем видно, що системи фільтрації є недоскональними та неточними. Кожен користувач має своє бачення ідеального житла, тому задовільнити всіх користувачів стандартизованими методами фільтрації неможливо.

Задачею є проектування та розробка системи інтелектуальної допомоги вибору житла з пропозицій.

1.3 Опис постановки задачі

На основі проведеного аналізу існуючих систем було посталено ряд вимог до системи пошуку, поміж стандартних фільтрів (ціна, площа, поверх, район):

- пошук житла за віддаленістю від станцій метро;
- пошук житла за віддаленістю до бізнес центрів, офісів;
- персоналізований пошук.

Основна мета створення інтелектуальної системи допомоги вибору житла – це покращення процесу пошуку найкращої пропозиції з великого об’єму даних за

рахунок використання інтелектуальних алгоритмів з одного боку та спрощення процесу надання житла в оренду за допомогою зручного інтерфейсу з іншого.

Призначення системи – це оптимізація та спрощення процесів оренди житла. А саме:

- оптимізація вибору житла для оренди;
- оптимізація формування пропозиції для оренди;
- спрощення процесу укладення угоди про оренду;
- спрощення процесу оплати оренди.

Виходячи з описаного вище, сформовано наступні задачі, які треба реалізувати для досягнення мети:

- розробка фільтрів для пошуку;
- розробка інтуїтивного інтерфейсу користувача;
- розробка алгоритмів сортування та виводу кращої пропозиції.

Висновок до розділу

Оглянуто проблематику систем оренди житла, описано процеси діяльності та предметну область. Визначені актори та їх функції, оглянуто бізнес процеси системи.

Проаналізовано існуючі системи, визначені їх переваги та недоліки. На основі аналізу встановлено вимоги до системи допомоги вибору житла для орендарів.

Сформовано **мету дослідження** – оптимізація процесу пошуку найкращої пропозиції з великого об’єму даних за рахунок використання інтелектуальних алгоритмів з одного боку та спрощення процесу надання житла в з іншого.

Для досягнення мети необхідно виконати наступні **завдання**:

- оптимізація вибору житла для оренди;
- оптимізація формування пропозиції для оренди;
- спрощення процесу укладення угоди про оренду;

– спрощення процесу оплати оренди.

2 МОДЕЛІ ТА МЕТОДИ

2.1 Формування моделі даних

У результаті аналізу предметної області та існуючих систем сформовано модель даних пропозиції оренди житла. Для повного опису пропозиції використовується наступні фактори, що достатньо описують пропозицію оренди:

- ціна;
- географічні координати;
- площа;
- тип квартири;
- новий будинок;
- поверх;
- опалення;
- ремонт;
- наявність меблів;
- сторона вікон;
- гілка метро.

Ціна є кількісною характеристикою та визначає величину місячної оплати. Оскільки пошук користувачем за діапазоном ціни є стандартним фільтром у багатьох системах дана характеристика є доцільною.

Площа – якісна оцінка оскільки користувач не шукає квартиру з визначеною площею, тому недоцільно використовувати чіткі фільтра для площі. Тому площа житла може мати наступні значення:

- мала;
- звичайна;
- простора.

Географічні координати є важливим параметром пропозиції оскільки за ними можна визначити рівень ціни серед схожих пропозицій. Географічними

координатами описується широтою та довготою. Розташуванням житла характеризує інфраструктуру мікрорайону, що впливає на ціну. Тому недоцільно порівнювати пропозиції які є суттєво віддаленими одна від одної.

Тип квартири є переліченням найпоширеніших форматів житла для оренди. Може приймати наступні значення:

- квартира студія;
- звичайна квартира;
- двоповерхова квартира;
- приватний будинок.

Новий будинок - це важлива характеристика для орендарів, оскільки, зазвичай, в нових мікрорайонах краще розвинена інфраструктура. Може приймати значення новий будинок або старий.

Поверх – є стандартним параметром при пошуку житла. Може приймати наступні значення:

- перший;
- останній;
- не перший;
- не останній;
- не перший і не останній.

Опалення може бути важливим параметром для більшості орендарів. Може приймати наступні значення:

- централізоване опалення;
- централізоване опалення з можливістю регулювання;
- окремий котел.

Ремонт – це параметр яким описується стан житла на момент оренди. Представлений варіантами:

- повний ремонт (новий);
- мінімальний (євро);
- відсутній.

Сторона вікон може бути важливим параметром для деяких користувачів яким важлива сонячна сторона чи тіньова сторона або напрямок вікон на певні сторони світу.

Гілка метро – це параметр, що є важливим для орендарів що користуються громадським транспортом. Значення залежать від міста.

2.2 Розробка методу розв’язання задачі

Для оцінки якості пропозицій необхідно отримати розбиття даних пропозицій на визначену кількість класів якості. Для даного розбиття має бути визначена певна функція міри подібності пропозицій за їх відношенню якості до ціни.

Проте вплив географічних координат на ціну значно перевищує вплив інших виділених критичних факторів зазначених вище. Тому доцільним є попередня кластеризація даних за географічними координатами, для подальшої класифікації в середині кластерів. Таким чином розбиття на класи стає значно більш чітким ніж розбиття всіх вхідних даних на класи без попереднього розбиття.

Тобто алгоритм складається з трьох етапів:

а) кластеризація по координатах;
б) формування функції міри відстані;
в) класифікація в кластерах.

2.3 Огляд методів розв’язання

2.2.1 Вибір алгоритму

Кластеризація (або кластерний аналіз) - це задача розбиття множини об'єктів на групи, які називаються кластерами [1]. Усередині кожної групи повинні виявитися «схожі» об'єкти, а об'єкти різних групи повинні бути якомога більш відмінні.

Застосування кластерного аналізу в загальному вигляді зводиться до наступних етапів:

- відбір вибірки об'єктів для кластеризації – дані пропозицій;
- визначення безлічі змінних, за якими будуть оцінюватися об'єкти у вибірці. При необхідності - нормалізація значень змінних;
- обчислення значень міри схожості між об'єктами;
- застосування методу кластерного аналізу для створення кластерів.

Алгоритми кластеризації поділяються на [2]:

а) ієрархічні і плоскі;

б) чіткі і нечіткі.

Ієрархічні алгоритми, які ще називаються алгоритмами таксономії, при розбитті вибірки на кластери, використовують не одне розбиття на неперехресні кластери, а будують систему вкладеного розбиття. Тобто на виході отримується дерево кластерів, де вся вибірка виступає у ролі кореня, а подальше розбиття - листя кореня.

На противагу, плоскі алгоритми розбивають вибірку даних лише один раз [3].

При використанні чітких алгоритмів при розбитті вибірки кожному об’єкту ставиться відповідний номер кластера, кожен об’єкт може належати лише одному кластері. Чіткі алгоритми, ще називають непересічними.

У разі використання нечітких алгоритмів (пересічних алгоритмів), при розбитті об’єктів по кластерам надають їм набір значень, що відповідає відношенню об’єкта до кожного кластеру з певною ймовірністю того, що об’єкт буду належати йому.

Ієрархічні алгоритми потребують метрик для визначення відстані між кластерами, так як кластери необхідно об’єднувати.

Існують такі метрики [2, 4]:

в) одиночний зв'язок (відстань найближчого сусіда);

г) повний зв'язок (відстань найбільш віддалених сусідів);

д) незважене попарне середнє;

е) виважена попарне середнє;

ж) незважений центроїдний метод;

з) зважений центроїдний метод (медіана).

Для цього методу характерно визначення відстані кластерів між двома сусідніми найбільш близькими об’єктами, об’єкти повинні знаходитися у різних кластерах. У результаті, такі кластери можна об’єднувати у ланцюги.

Проте для задачі розбитті по параметру – географічні координати, такий тип не є доцільним, оскільки координати розподілені по площини, а не згруповані у ланцюжки.

На противагу першій метриці, у цьому випадку відстань вимірюється між двома найбільш віддаленими об’єктами, які знаходяться у різних кластерах. Перевага цього методу в тому, що він показує гарні результати, якщо об’єкти знаходяться у різних групах. Якщо кластер має видовжену форму та складає ланцюжок, такий метод не є придатним, так як результати не відповідають очікуванням.

Для задачі поставленій в системі інтелектуального пошуку житла дана оцінка є прийнятною. Однак, при розбитті може виникнути ситуація, коли розмір кластерів суттєво відрізняється один від одного, що може пливати на дану метрику.

Для задачі поставленій в системі інтелектуального пошуку житла дана оцінка є прийнятною. Однак, при розбитті може виникнути ситуація, коли розмір кластерів суттєво відрізняється один від одного, що може пливати на дану метрику.

Для даного методу відстань між двома кластерами вираховується як середня відстань між кожною парою об’єктів у цих кластерах. Є доцільним, коли об’єкти сформовані у групи, проте метод також добре працює, якщо кластери утворюють ланцюги.

Для кластеризації по координатах дана оцінка не є бажаною, оскільки очікується суттєва різниця в розмірах кластерів.

Метод виваженого попарного середнього при обчисленнях використовує кількість об’єктів, що містяться всередині кластеру як ваговий коефіцієнт. Тобто даний метод слід використовувати якщо очікується суттєва різниця в розмірах результуючих кластерів.

Для даної системи така оцінка є доцільною, оскільки групи за координатами можуть мати різний розмір в залежності від розподілення пропозицій по мапі.

Метод незваженої центроїдної відстані використовує евклідову відстань між центрами кластерів як міру відмінності кластерів.

Дана оцінка підходить для кластеризації по координатах, проте є чутливою до форми та розміру кластерів.

Метод зваженої центроїдної відстані є аналогічним до незваженого центроїдного, але має міру ваги що відповідає до розмірів кластеру. Відповідно якщо очікується суттєва різниця в розмірах результуючих кластерів слід застосувати саме зважений центроїдний метод.

Дана оцінка підходить для кластеризації по координатах, проте є чутливою до форми кластерів.

Виходячи з аналізу метрик відстані між кластерами для кластеризації по координатах використовується виважена попарна середня оцінка відстані.

Проаналізуємо алгоритми кластеризації [5]:

а) алгоритми ієрархічної кластеризації;
б) алгоритми квадратичної помилки;
в) алгоритми, засновані на теорії графів;
г) алгоритм виділення зв'язкових компонент.

Алгоритми ієрархічної кластеризації класифікують на дві групи:

- низхідні алгоритми
- висхідні алгоритми.

Більш популярними є висхідні алгоритми. Основний принцип є в розміщенні кожного елемента в окремий кластер, які потім об’єднуються в більші кластери, доки всі елементи не стануть елементами одного кластеру. Таким чином будується вкладене розбиття.

Низхідні алгоритми мають такий самий принцип що і висхідні, але на початку алгоритму всі елементи є членами одного кластеру, який розбивається на більш дрібні кластери.

Результати ієрархічних алгоритмів представляють у вигляді дерева розбиття.

Для обчислення відстаней між кластерами використовують відстань за одиночним зв'язком або повним зв'язком.

Проте для даної задачі система повного розбиття, є зайвою в контексті розв'язуваної задачі, оскільки не підходить для оцінки відстані для очікуваного розбиття кластерів.

Задача кластеризації може бути представлена як задача мінімізації середньоквадратичної помилки розбиття на кластери.

Найпоширенішим алгоритмом квадратичної помилки для кластеризації є метод к-середніх [6], що відноситься до типу плоских алгоритмів. Цей алгоритм вимагає зазначене число кластерів, які будуть розбиті з найбільшою відстанню один від одного. Алгоритм складається з декількох етапів:

1) Вибрано k випадкових точок, які є початковими «центрами мас» кластерів.
2) Кожен об'єкт відноситься до кластеру з найближчим «центром мас».
3) Перераховуються «центри мас» кластерів відповідно до їх поточного складу.

Якщо критерій зупинки алгоритму не задоволений, повернутися до п. 2.

Як критерій зупинки роботи алгоритму зазвичай вибирають мінімальну зміну середньо-квадратичної помилки. Так само можна зупиняти роботу алгоритму, якщо на кроці не було об'єктів, що перемістилися з кластера в кластер.

До недоліків даного алгоритму можна віднести необхідність задавати кількість кластерів для розбиття, що є суттєвим для даної задачі, проте даний алгоритм має перевагу в імплементації, оскільки не потребує збереження матриці відношень в

пам’яті програми. Недолік з кількістю кластерів можна вилучити модифікацією алгоритму.

Алгоритми, засновані на теорії графів полягають в представленні об’єктів у вигляді графа, де вершини – це об’єкти, а ребра – вага, рівна відстані між об’єктами. Перевагою таких алгоритмів кластеризації є наочність розбиття і можливість внесення різних удосконалень, заснованих на геометричних принципах. Основними алгоритмами є алгоритм виділення зв'язкових компонент, алгоритм побудови мінімального покриваючого дерева і алгоритм пошарової кластеризації.

Недоліком даного алгоритму є необхідність збереження всього графу відношень в пам’яті програми, що обмежує розмір вибірки даних для кластеризації, що є неприпустимо для даної системи.

В алгоритмі виділення зв'язкових компонент задається вхідний параметр R і в графі видаляються всі ребра, для яких «відстані» більше R. Сполученими залишаються тільки найбільш близькі пари об'єктів. Сенс алгоритму полягає в тому, щоб підібрати таке значення R, що лежить в діапазон всіх «відстаней», при якому граф «розвалиться» на кілька зв'язкових компонент. Отримані компоненти і є кластери.

Для підбору параметра R зазвичай будується гістограма розподілів попарних відстаней. У завданнях з добре вираженою кластерної структурою даних на гістограмі буде два піки - один відповідає внутрішнім кластерним відстаням, другий – між кластерним відстаням. Параметр R підбирається із зони мінімуму між цими піками. При цьому управляти кількістю кластерів за допомогою порога відстані досить важко.

Даний алгоритм також має проблеми з об’ємом пам’яті при роботі, тому не підходить для задачі кластеризації великого об’єму даних.

Для остаточного визначення алгоритму для кластеризації по координатах порівняємо складність алгоритмів (таблиця 2.1).

Таблиця 2.1 - Складність алгоритмів

Алгоритм кластеризації	**Обчислювальна складність**
Ієрархічний	$O(n^2)$
k-середніх	O(nkl), де k – кількість кластерів, l – кількість ітерацій
c-середніх	
Виділення зв’язкових компонент	Залежить від алгоритму
Мінімальне покриваюче дерево	$O(n^2 \log n)$
Пошарова кластеризація	O(max(n, m)), де $m < n(n - 1)/2$

У результаті аналізу обрано алгоритм квадратичної помилки k-середніх, але з модифікацією для визначення оптимальної кількості кластерів, оскільки даний алгоритм найкраще підходить для даної задачі та має лінійну складність, що не залежить від об’єму даних, та не потребує значної кількості пам’яті для обчислення програмою.

2.4 Розробка алгоритму обробки даних пропозицій

2.4.1 Алгоритм кластеризації

Перший етап обробки даних пропозицій – кластеризація за координатами. Для даної задачі обрано алгоритм к-середніх. Оскільки планується обробка великого обсягу даних програмними засобами, тому алгоритм має відповідати наступним критеріям:

- лінійна складність алгоритму;
- алгоритм не повинен зберігати в пам’яті всю матрицю або граф даних;
- алгоритм не потребує визначеної кількості кластерів.

Алгоритм к-середніх повністю задовольняє зазначені критерії, проте потребує визначену кількість кластерів.

Оскільки оптимальна кількість кластерів є не визначеною то виконується додатковий етап в алгоритмі для її визначення [7]. Виконуються проходи алгоритму

для різної кількості кластерів. При кожному проході алгоритму визначається середньо квадратична похибка розподілення

$$e^2 = \sum_{j=1}^{K} \sum_{i=1}^{n_j} \|x_{ij} - c_j\|^2,$$

де x – елемент кластеру, тобто параметр пропозиції оренди в даному випадку;

K – кількість кластерів;

c – централи кластерів.

Коли при зміні кількості кластерів значення похибки змінюється на незначну величину – то таку кількість кластерів можна вважати оптимальною.

Очевидно, такий алгоритм виконувати на кожен запит кожного орендаря є недоцільним, тому перерозподіл кластерів може виконуватись з певною періодичністю, незалежно від запитів користувача.

2.4.2 Формування оціночної функції

Для виконання алгоритму класифікації необхідно визначити оціночну функцію для визначення подібності пропозицій.

Дані пропозицій можуть бути представлені у вигляді лінійної моделі, до якої можуть бути застосовані методи лінійної регресії [8]. Для даної системи використано евклідова відстань між точками у багатовимірному просторі.

Критерієм для порівняння пропозицій є їх відносна якість, яка визначається

$$q(P, C) = \sum_{i=1}^{n} P_i C_i,$$

де P – вектор нормованих параметрів пропозицій;

C – вагові коефіцієнти параметрів.

Вектор вагових коефіцієнтів визначається з урахуванням впливу параметрів, що вказані користувачем як критичні.

Тоді якість відносно ціни

$$q_r(P, C) = \frac{q(P, C)}{P_{\text{ціна}}}. \quad (2.1)$$

Для формування оціночної функції для алгоритму класифікації використано найпоширенішу міру відстані – евклідову відстань

$$p(x, x') = \sqrt{\sum_{i=1}^{n} (x_i - x_i')^2}, \quad (2.2)$$

де x та x′ - це елементи n вимірного простору, що описують дві точки між якими визначається відстань.

Поєднавши міру подібності (2.1) та евклідову міру відстані (2.2) отримаємо

$$p(P, P', C) = \sqrt{\sum_{i=1}^{n} \frac{(P_i - P'_i)^2 {C_i}^2}{\left(P_{\text{ціна}}^{(i)} - P'^{(i)}_{\text{ціна}}\right)^2}}. \quad (2.3)$$

де P, P′ - це порівнювані пропозиції оренди, що представлені векторами параметрів;

C – вектор вагових коефіцієнтів.

2.4.3 Алгоритм класифікації

Останнім етапом в роботі алгоритму є класифікація. Необхідно розподілити дані в кожному кластері на попередньо визначену кількість класів.

Класи є попередньо визначені. Для кожного класу якості сформовано відповідну пропозицію, що буде еталоном класу.

Для даної задачі обрано метод c-середніх [9], що є нечітким методом кластеризації. Оскільки чітке розподілення є не обов’язковим і межі між класами будуть не чіткими. Тому ймовірнісна оцінка класу якості пропозиції оренди є доцільною.

Вагомим аргументом при виборі методу с-середніх є його лінійна складність та відсутність потреби зберігання всієї матриці розподілення в пам’яті програми при імплементації алгоритму.

Для даної задачі алгоритм модифікований, оскільки централи вибираються не випадково, а є визначеними еталонами класу.

Алгоритм виконується доки критерій нечіткої похибки (2.4) суттєво змінюється [10].

$$E^2(X,U) = \sum_{i=1}^{N}\sum_{k=1}^{K} U_{ik} \left\| x_i^{(k)} - c_k \right\|^2, \tag{2.4}$$

де X – матриця даних пропозицій;

U – матриця відношень пропозицій до кластерів;

K – кількість кластерів;

N – кількість параметрів моделі пропозиції оренди.

У результаті роботи алгоритму маємо розбиття даних про пропозиції оренди за класами якості, з урахуванням впливу вибору користувача критичних факторів.

Виконання даного алгоритму класифікації на кожен запит користувача є доцільним, оскільки виконується попередня фільтрація вхідного масиву даних пропозицій за критичними значеннями параметрів вказаних користувачем. Також виконується фільтрація за географічними координатами, тобто відсіються географічні кластери, що значно зменшує вхідний об'єм даних.

Висновок до розділу

Розроблено метод розв’язання задачі та сформовано модель даних. Оглянуто алгоритми розв’язання та визначено які алгоритми будуть використані у системі.

Розроблено новий алгоритм для розв’язання поставленої задачі. Сформовано математичну модель системи.

3 ОПИС ПРОГРАМНОГО ТА ТЕХНІЧНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Засоби розробки

3.1.1 Інтегроване середовище розробки

Visual Studio Code поєднує простоту редактора вихідного коду з потужним інструментом для розробників, таким як IntelliSense завершення коду та засоби налагодження (Debug).

В основі програми Visual Studio Code є швидкий редактор вихідного коду, який ідеально підходить для щоденного використання. Завдяки підтримці сотень мов, VS підвищує продуктивність завдяки підсвічуванням синтаксису, відповідності дужок, автоматичним відступом, вибором коробки, фрагментами тощо. Інтуїтивні комбінації клавіш, просте налаштування та відображення комбінацій клавіш для клавіатури, дозволяють легко переміщувати код.

Для серйозного кодування часто користуються інструментами з більш зрозумілим кодом, ніж просто блоки тексту. Код Visual Studio включає вбудовану підтримку для завершення коду IntelliSense, багате розуміння семантичного коду та навігацію та рефакторинг коду.

Налагодження часто є однією з особливостей, яку розробники пропускають найбільше в процесі короткого кодування. Visual Studio Code включає інтерактивний відладчик, тому можливо проходити по коду, перевіряти змінні, переглядати стеки викликів і виконувати команди в консолі.

VS також інтегрується з інструментами побудови та створення сценаріїв для виконання часто повторюваних завдань, що сприяє швидшому повсякденному робочому процесу. VS Code має підтримку Git, тому можливо працювати з керуванням джерелами, не виходячи з редактора, включаючи перегляд очікуючих змін.

VS включає збагачену вбудовану підтримку розробки Node.js з JavaScript та TypeScript, що працює на тих же базових технологіях, що і Visual Studio. VS також

включає чудові інструменти для веб-технологій, таких як JSX / React, HTML, CSS, SCSS, Less та JSON.

В архітектурному відношенні Visual Studio Code поєднує в собі найкращі веб-, рідні та мовні технології. Використовуючи Electron, VS Code поєднує в собі веб-технології, такі як JavaScript та Node.js, зі швидкістю та гнучкістю нативних програм. VS Code використовує новішу, більш швидку версію того самого промислового редактора, заснованого на HTML, який використовував хмарний редактор "Монако", інструменти F12 Internet Explorer та інші проекти. Крім того, VS Code використовує архітектуру сервісу інструментів, яка дозволяє їй інтегруватися з багатьма тими ж технологіями, що і Visual Studio, включаючи Roslyn для .NET, TypeScript, двигуна налагодження Visual Studio тощо.

Visual Studio Code включає в себе модель публічної розширюваності, яка дозволяє розробникам створювати та використовувати розширення та рясно налаштовувати їхній досвід редагування-збирання-налагодження.

3.2 Архітектура програмного забезпечення

Система має виконувати декілька основних функцій:

- обробляти HTTP запити;
- кластеризувати дані;
- класифікувати дані;
- зберігати дані.

Функції обробки HTTP запитів, кластеризації даних та класифікації даних є досить складними в імплементації та мають складну структуру, тому є сенс розділити наведені функції на окремі мікро сервіси.

Система складається з трьох частин (рисунок 3.1):

- клієнт;
- мікро сервіси;
  a) веб сервер;

б) сервіс кластеризації;

в) сервіс класифікації;

– база даних.

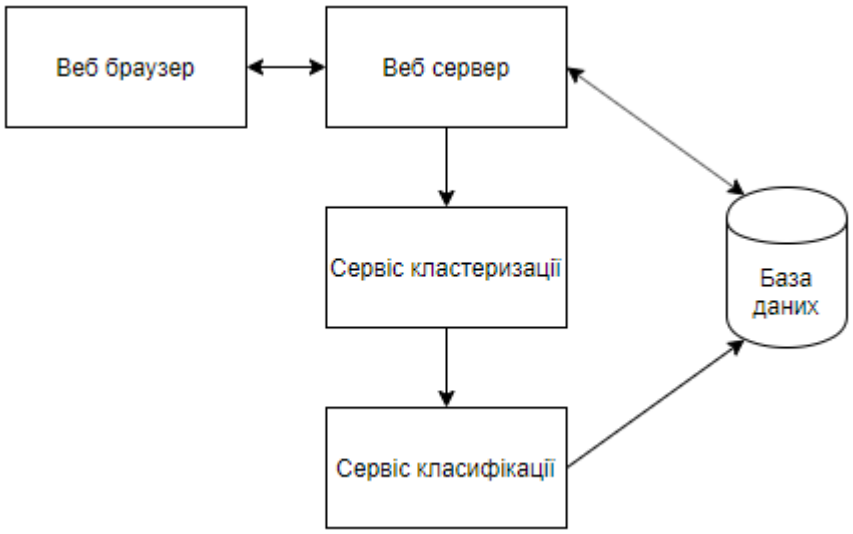


Рисунок 3.1 – Архітектура системи

3.2.1 База даних

У проекті мають бути збережені наступні групи даних:

– дані користувачів;
– пропозиції;
– результати роботи інтелектуальних алгоритмів.

Дані користувачів мають бути надійно збережені, тому є наступні варіанти задля їх збереження:

– в основній базі даних з базою пропозицій;
– в окремій базі даних тільки з обліковими записами;
– у хмарному сервісі збереження даних.

Облікові записи користувачів потребують сховище високої надійності з низькою ймовірністю проведення хакерської атаки з метою вилучення особистих даних користувачів.

Всі дані окрім облікових записів користувачів є публічними. Тому не потребують особливого захисту. Має бути збережена тільки цілісність даних, з чим справляється будь яке рішення з імплементації бази даних.

Створення окремої бази даних для облікових записів є не раціональним та більш складним для імплементації ніж використання хмарних сервісів. Найпростішим для інтеграції проте надійним сервісом авторизації є Firebase.

Firebase надає сервіси, прості у використанні SDK та готові бібліотеки інтерфейсів для автентифікації користувачів. Firebase підтримує автентифікацію за допомогою паролів, номерів телефонів, популярних федеральних постачальників ідентифікаційних даних, таких як Google, Facebook та Twitter тощо.

Автентифікація Firebase тісно інтегрується з іншими службами Firebase і використовує такі галузеві стандарти, як OAuth 2.0 та OpenID Connect, тому вона може бути легко інтегрована у окремий сервер.

Для пропозицій необхідна база даних що є здатною працювати з великими об’ємами даних та складними індексами. Структура пропозиції є визначеною. Для пропозицій можливі наступні варіанти моделей баз даних:

а) реляційні бази даних;
б) база даних NoSQL;
в) графові бази даних;
г) сховище типу ключ – значення;
д) сховище широких колонок.

Реляційні бази даних зберігають дані у вигляді рядків таблиці з визначеними стовбцями. Кожен стовпець має визначений тип.

Тобто реляційні бази даних потребують заздалегідь визначеної структури даних, що може бути збережена у вигляді рядків.

Для даної системи необхідно збереження пропозицій – що мають визначену структуру та результатів роботи алгоритмів, а саме результат класифікації, що є відношенням пропозицій до класів якості.

Структура реляційних баз даних повністю задовольняє потреби системи.

Реляційні бази даних - найпоширеніший і широко використовуваний вид, з поважних причин. Модель, хоч і негнучка, є надійною, уникає дублювання даних і дає можливість розробникам працювати із збереженням і передбачуваними результатами. Крім того, ця категорія містить багато зрілих доступних баз даних, таких як PostgreSQL або MySQL - і зрілість, і отримана стабільність та безпека, мабуть, є найважливішим фактором при виборі будь-якої бази даних.

Окрім деяких винятків, NoSQL бази даних зберігають всю інформацію, що стосується певного об'єкта (наприклад, пропозиції оренди) в єдиному, неструктурованому фрагменті даних JSON, який називається "об'єктом" або "документ". Кілька однотипних документів розташовуються в "списках" або "колекціях". Об'єктні бази даних сильно відрізняються способом створення, зчитування та пошуку даних, починаючи від API RESTful (CouchDB) і закінчуючи функціональними поняттями програмування, такими як Map-Reduce (MongoDB).

Гнучкість, що випливає з просто перекиданням різних структурованих документів у одну колекцію, не потрібно турбуватися про типи даних та міцні зв’язки та залишати це для СУБД.

Для даної системи дана гнучкість дозволяє поєднати дані класифікації з даними пропозиції, що дозволяє зменшити складність запиту до БД для отримування вибірки пропозицій за вподобаннями користувача. Проте відсутність примусового виконання БД в об'єкті повинно компенсуватися дисципліною розробників та пильністю. Крім того, високореляційні дані все ще краще зберігаються у реляційній або графічній базі даних, лише підмножина випадків використання з досить ізольованими наборами даних є гарною підходящою для моделі документа / колекції. І дані пропозиції є високореляційними, тому NoSQL не є кращим варіантом для даної системи.

Графові БД пропонують інший спосіб мислення щодо взаємозв'язку даних і можуть запропонувати великий компроміс між суворими таблицями та розсипчастими документами. У графічній базі даних кожна сутність (вузол) є ізольованим документом, що містить список нестабільних властивостей. Ці вузли з'єднані "краями", які задають їх співвідношення. Краї можуть мати властивості самостійно.

Більшість графових БД постачаються з багатими наборами інструментів для запиту, обходу та оцінки складних мереж вузлів та ребр, наприклад. щоб знайти всі підключені вузли, вузол з найбільшою кількістю з'єднань, найближчий сусід тощо.

Графові БД - це чудовий спосіб структурувати взаємопов'язані дані. Проте дана система має досить просту структуру даних, а для більш простих випадків використання графових БД може мати досить багато накладних витрат. Крім того, мислення в графіках - це, мабуть, найскладніша модель у цьому списку, що створює додаткові навантаження на розробника.

Сховища типу ключ - значення - це швидкий спосіб збереження деяких даних, ідентифікованих ключем. Їх простота робить доступ дуже швидким, що робить їх чудовими придатними для нереляційних даних.

Проте дані є реляційними, мають складні структури та потребують запитів і пошуку – тому дане рішення не доцільне.

Сховище широких колонок – це якщо ключ-значення зберігається з другим виміром, або з будь-якою кількістю динамічних стовпців та рядків. Сховище широких колонок дозволяють швидко знаходити рядки, стовпці або їх перетин і дозволяють швидко записувати чи читати величезну кількість даних.

Якщо можливо організувати всі дані в одному гігантському аркуші "Excel" то дане рішення підходить для системи. Проста модель даних сховища широких колонок надає простоту масштабування.

Хоча деякі сховища, такі як Apache Cassandra, підтримують основні запити, функції вищого рівня, такі як агрегати (суми, рахунки, середні показники), все ще

можуть бути недоступними. Крім того, багато сховищ забезпечують часткову стабільність, що робить їх поганим вибором для транзакційних застосувань.

Дана система потребує більш складної структури даних та підтримку транзакцій, тому сховище широких колонок не підходить під критерії.

З найпопулярніших реляційних баз даних є вибір між MySQL та PostgreSQL.

Історично, MySQL мав репутацію надзвичайно швидкої бази даних для великих робочих навантажень операціями читання, іноді ціною паралельності операцій, коли їх змішували з операціями запису.

PostgreSQL, також відомий як Postgres, рекламує себе як "найсучаснішу реляційну базу даних з відкритим кодом у світі". Він був розроблений для збагачення функцій, розширення та відповідності стандартам. У минулому продуктивність Postgres була більш врівноваженою - читання, як правило, повільніше, ніж MySQL, але він міг записувати великі обсяги даних ефективніше, і справлявся з паралельністю краще.

Відмінність продуктивності між MySQL та Postgres значною мірою стерта в останніх версіях. MySQL все ще дуже швидко читає дані, але лише за умови використання старого механізму MyISAM. Якщо використовувати InnoDB (що дозволяє транзакції, ключові обмеження та інші важливі функції), відмінності незначні (якщо вони навіть існують). Ці функції абсолютно важливі для для даної системи, тому використання старого механізму не є можливим. З іншого боку, MySQL також був оптимізований для зменшення розриву, коли мова йде про великі об’єми запису даних.

Вибираючи між MySQL і PostgreSQL, продуктивність не повинна бути фактором вона буде задовільною в будь-якому випадку, навіть якщо планується значне масштабування системи. Обидві платформи чудово здатні до реплікації, і багато хмарних провайдерів пропонують керовані масштабовані версії будь-якої з цих баз даних. Тому варто розглянути інші переваги Postgres над MySQL.

Postgres - об’єктно-реляційна база даних, тоді як MySQL - суто реляційна база даних. Це означає, що Postgres включає такі функції, як успадкування таблиць та перевизначення функцій, що може бути важливим. Postgres також більш дотримується стандартів SQL.

Postgres обробляє паралельність краще, ніж MySQL з кількох причин:

- Postgres реалізує мультиверсійний контроль сумісності без блоків читання;
- Postgres підтримує паралельні плани запитів, які можуть використовувати декілька процесорів / ядер;
- Postgres може створювати індекси не блокуючим способом (через синтаксис CREATE INDEX CONCURRENTLY) і може створювати часткові індекси (для Наприклад, якщо у вас є модель з м'якими видаленнями, ви можете створити індекс, який ігнорує записи, позначені як видалені);
- Postgres відомий тим, що захищає цілісність даних на рівні транзакції. Це робить його менш вразливим до пошкодження даних.

Установка Postgres за замовчуванням зазвичай працює краще, ніж за замовчуванням MySQL.

Postgres є дуже розширюваним. Він підтримує ряд розширених типів даних, недоступних у MySQL (геометричні / ГІС, типи мережевих адрес, JSONB, які можуть бути індексовані, нативний UUID, часові позначки, відомі часовому поясу). Якщо цього недостатньо, можливе також додавання власних типів даних, операторів та типів індексів.

Postgres є справді з відкритим кодом, тоді як у MySQL виникли деякі проблеми з ліцензуванням. Він був запущений як продукт компанії (з безкоштовною та платною версією), а придбання Oracle MySQL AB у 2010 році призвело до певних проблем серед розробників щодо його майбутнього статусу з відкритим кодом. Однак існує декілька вилок з відкритим кодом оригінального MySQL (MariaDB, Percona тощо), тому наразі це не вважається великим ризиком.

Незважаючи на всі ці переваги, все ж є деякі невеликі недоліки використання Postgres, які слід врахувати:

- Postgres все ще менш популярний, ніж MySQL, тому є менша кількість сторонніх інструментів для розробників / адміністраторів баз даних;
- Postgres розгортає новий процес для кожного нового клієнтського з'єднання, який виділяє нетривіальний об'єм пам'яті (близько 10 МБ);
- Postgres побудований з фокусом на розширюваність, відповідності стандартам, масштабованості та цілісності даних - іноді за рахунок швидкості. Тому для простих, важких для читання робочих процесів Postgres може бути гіршим вибором, ніж MySQL.

Проте для даної задачі Postgres є більш збалансованим рішенням оскільки надає ряд необхідних функцій яких не вистачає у MySQL.

Тому для збереження інформації системи про пропозиції та дані результатів виконання інтелектуальних алгоритмів обробки даних в системі буде використано Postgres. А для надійного збереження облікових записів користувачів буде використано хмарний сервіс Firebase.

Для вирішення поставлених задач сформовано структуру бази даних (рисунок 3.2).

База даних є реляційною, тому дані структуровані в таблиці з відношеннями між ними.

Структура БД є досить простою оскільки основна задача системи є робота з великими об’ємами однотипних даних.

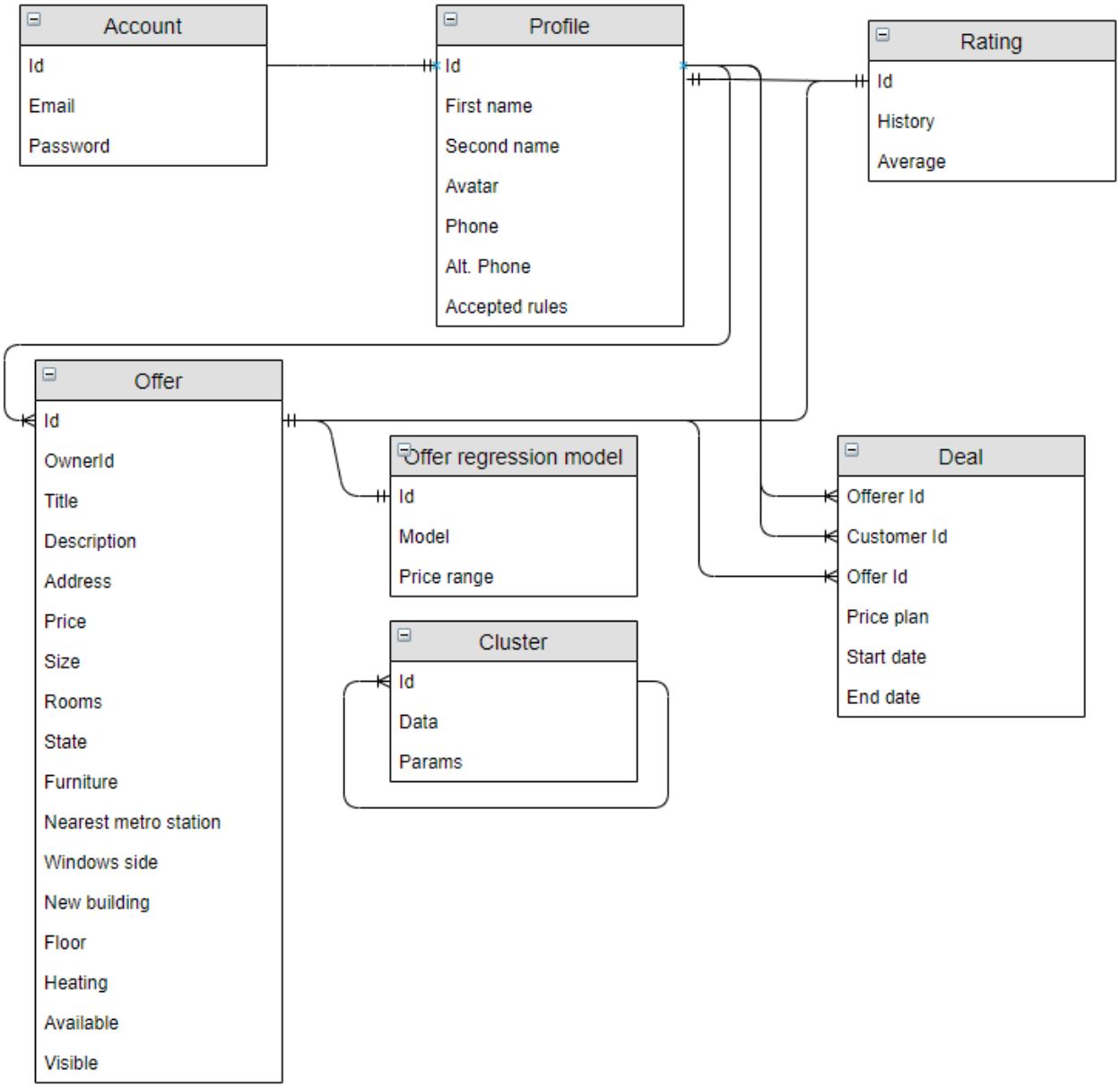


Рисунок 3.2 – Концептуальна схема бази даних

Таблиця Account – це таблиця зареєстрованих користувачів у модулі Firebase Authorization.

Таблиця Profile – це таблиця, що розширює таблицю Account додатковою інформацією про користувача. Має відношення один до одного до таблиці Account.

Таблиця Offer – це таблиця пропозицій оренди. Має відношення багато до одного до таблиці Profile.

Таблиця Rating – це таблиця рейтингу користувачів. Має відношення один до одного до таблиць Profile та Offer.

Таблиця Deal зберігає інформацію про акти оренди житла. Має відношення багато до одного до таблиць Profile та Offer.

Таблиця Cluster є результатом роботи алгоритму кластеризації та являє собою дерево кластерів де пропозиції згруповані за координатами та класами якості. Ця таблиця не має явного зв’язку з таблицею пропозицій, оскільки дані кластеризації є дійсними на момент роботи алгоритму, після чого цілісність даних кластеризації не гарантується [11].

3.2.2 Серверна частина

Для імплементації серверної частини системи було розглянуто декілька рішень:

а) Express.js;
б) Meteor Js;
в) Nest Js;
г) Adonis Js.

Express.js - це швидкий, неосуджений, мінімалістичний веб-фреймворк для Node.js. Це просто технологія, побудована на Node.js, яка поводиться як проміжне програмне забезпечення, яке допомагає керувати серверами та маршрутами. Якщо дивитися на асинхронний характер Node.js і те, що Express.js був побудований на вузлі, можливість побудови легкої програми, яка може обробляти більше, ніж один запит, насправді залежить від можливості обслуговування технологій, таких як express.

Цей надійний API дозволяє користувачам налаштовувати маршрути для надсилання / прийому запитів між передньою частиною та базою даних (виконуючи функцію HTTP-сервера). Хорошою перевагою express є те, що він підтримує безліч інших пакетів та інших двигунів шаблонів, таких як Pug, Mustache, EJS та багато іншого.

Деякі з численних переваг Express.js включають:

- майже стандарт для веб-програмного забезпечення Node.js;
- повністю настроюється;
- крива низького рівня навчання;
- основна увага зосереджена на веб-переглядачах, що робить шаблони майже вбудованою функцією.

Express.js з часом показав, що його популярність вартує ажіотажу своїми простими у використанні методами та функціями. Це, мабуть, найпопулярніший фреймворк Node.js, доступний для спільноти JavaScript на GitHub із понад 41 000 зірок (Github stars: 41,036).

Документація Meteor визначає метеор як повноцінну платформу JavaScript для розробки сучасних веб і мобільних додатків. Основна перевага - це оновлення в режимі реального часу. Після внесення змін у веб-додаток він автоматично оновлює шаблон із останніми змінами.

Рамка Node.js робить розробку досить спрощеною, забезпечуючи платформу для того, щоб весь додаток був побудований на одній мові: JavaScript. Змусити його функціонувати настільки ж ефективно як на сервері, так і на стороні клієнта.

Метеор виступає перед конкурентами можливістю обслуговувати великі проекти, такі як реакційна комерція.

Найбільш захоплюючим аспектом Meteor є дуже багата та організована документація / велике співтовариство, яке він має, допомагаючи користувачам швидко навчатися.

Той факт, що метеор використовує набір даних Facebook GraphQL для розробки Meteor Apollo ще в 2016 році, лише свідчить про те, що у них є гарні плани та сприйняття того, що має бути в майбутньому для баз даних, як ними керувати і контролювати.

NestJs - це структура, побудована разом з Node.js, вона використовується для створення ефективних масштабованих додатків на сервері Node.js. Nest використовує прогресивний JavaScript і пишеться з TypeScript. Будучи побудованим

за допомогою TypeScript, це означає, що Nest має сильну типізацію та поєднує в собі елементи OOP (об'єктно-орієнтоване програмування), FP (функціональне програмування) та FRP (функціональне реактивне програмування).

Nest також використовує Express, він пропонує архітектуру додатків, що дозволяє легко створювати додатки, які легко перевіряються, масштабуються, легко поєднуються та легко ремонтуються.

AdonisJs - це фреймворк Node.js. З офіційної документації: "AdonisJs - це фреймворк MVC Node.js, яка працює на всіх основних операційних системах. Він пропонує стабільну екосистему для написання веб-програми на стороні сервера, щоб зосередитись на потребах бізнесу замість приймання рішення який пакет вибрати або ні".

AdonisJs має підтримку ORM, створена з урахуванням баз даних SQL (PostgreSQL). Він створює ефективні SQL-запити і заснований на ідеї активного запису (Active record). Його конструктор запитів легко засвоїти і дозволяє швидко будувати прості та складні запити.

AdonisJs має хорошу підтримку для баз даних NoSQL, як MongoDB.

Оскільки задачею розробки серверної частини є побудова API та взаємодія з базою даних PostgreSQL то очевидним вибором є Adonis JS як найпопулярніший фремворк станом на 2019 рік.

3.2.2 Клієнтська частина

Домінування JavaScript у світовому Інтернеті очевидно і не викликає сумнівів.

Однією з головних причин цього є загальна тенденція переміщення розробників бекенду в інтерфейс і перехід від веб-додатків на стороні сервера до клієнта.

JavaScript є універсальним рішенням та оптимальним для багатьох розробок. Він може працювати як на клієнті, так і на сервері (з NodeJS), що відкриває новий спосіб спільного використання коду між фронтом та бекендом. Крім того, він швидкий, масштабований, порівняно простий у навчанні.

Проблема з вибором технології постає через те, що використання нативного JS є трохи старомодною і часто контрпродуктивною. Натомість з'явилися нові бібліотеки та фреймворки, - які допомагають зробити завдання розробників JS швидшими та простішими, хоча вибір потрібного не є ні швидким, ні простим.

Великою проблемою постає, обрання фреймворку чи бібліотеки. Фреймвок - це скелет програми. Він містить як конкретні, готові до використання елементи (наприклад, бібліотеки та сценарії), так і загальну схему того, як слід ними користуватися.

Бібліотека - це набір функцій, які програма може викликати для виконання певного завдання.

Найпоширенішими в даний час в рамках JavaScript є React, Angular і Vue – останній є найзручнишим та найдоступнішим, але за Angular є Svelte. Немає згоди щодо того, чи можна його навіть назвати фреймворком, оскільки він є скоріше компілятором, а значить, його значення засноване на зменшенні кількості коду. Svelte дозволяє записувати компоненти, використовуючи лише HTML, CSS, JavaScript, і під час збірки він збирає їх у невеликі модулі JavaScript.

Наразі Svelte може використовуватися рідко, але оскільки це робить додатки помітно простішими.

Але чи означає це, що Vue є найкращим вибором? Так, якщо мова йде про доступність. Ця відносно нова структура була розроблена в 2014 році, але лише за півтора року вона стала стандартом. Його використовують такі гіганти, як Алібаба, Байду і Тенсент, які виявили, що Vue є стабільним і надійним. Крім того, Vue зазвичай вважається найпростішим у навчанні та найзручнішим у використанні через повернення до своїх коренів JS.

Такі переваги надають розробникам ширші можливості HTML, CSS та JavaScript, і для багатьох з них це більш природний спосіб створення веб-додатків, ніж використання JSX, хоча також дозволяє використовувати JSX завдяки внутрішнім плагінам.

У програмі React це дозволяє розробникам писати JavaScript, схожий на HTML. Можна використовувати React без JSX, але JSX дозволяє React показувати більше корисних повідомлень про помилки та попередження.

Крім того, Vue має детальну документацію, що значно згладжує криву навчання для нових розробників Vue, що робить технологію чудовою придатною для проектів в основному будь-якого типу.

Vue - який сам по собі невеликий - ідеально підходить для створення односторінкових додатків, але також підходить для більш складних реалізацій. Внутрішні елементи можна інтегрувати в інфраструктуру без шкоди для продуктивності.

До переваг Vue можна віднести:

- доступність;
- стабільність;
- надійність;
- простота;
- документація.

Angular - це найвимогливіший і найскладніший JS, але він також чудово підходить для корпоративних проектів. Це дає цілу екосистему інструментів, особливостей та можливостей. Його архітектура (особливо TypeScript) забезпечує хорошу якість коду, що корисно великим командам, оскільки опускається велика кількість помилок, але гнучкість цього фреймворку є обмеженою.

У Angular не потрібно покладатися на сторонні бібліотеки для створення складних, динамічних додатків - він забезпечує майже все необхідне та, швидше за все, більше, але реалізацію потрібно робити конкретно для Angular.

Основна технічна відмінність Angular від React або Vue полягає в тому, що Angular змушує нас використовувати TypeScript (набір Javascript), що може бути складно для тих розробників, які не знайомі з такими поняттями, як типи, класи та інтерфейси.

До переваг можна віднести:

- розробка корпоративних проектів;

- зменшення кількості помилок.

До недоліків відносяться:

- додаткове вивчення TypeScript;
- менша гнучкість.

Сам React технічно є більшою частиною бібліотеки, але міцна система, побудована навколо нього, змушує працювати як повноцінна структура. React був розроблений Facebook у 2013 році і тоді був величезним, оскільки, порівняно з Angular, він здавався дуже легким, швидким та гнучким. React одразу став улюбленцем розробників JavaScript, оскільки це дозволило їм витрачати більше часу на написання JS-коду, ніж специфічний для фреймворку код.

Основною перевагою React була його віртуальна реалізація DOM. Локальна копія HTML DOM React, локальне порівняння елементів, дозволяє бібліотекам React відображати лише фактично змінені компоненти та економити час, уникаючи багатьох зайвих операцій.

На сьогодні React - це найпопулярніший фреймворк JS, який використовують Facebook, Netflix, Yahoo, Codecademy, Dropbox, Airbnb, Asana, Microsoft, Slack та багато іншого. Тому очевидним вибором для розробки клієнтської частини є React.

Висновок до розділу

Сформовано архітектуру програмного додатку, описано засоби розробки.

Проаналізовано типи баз даних та виконано порівняльний аналіз, у результаті якого вибрано СУБД PostgreSQL, як базу даних, AdonisJs, як веб сервер та основу для мікро сервісів, та ReactJs, як бібліотеку для клієнтської частини.

4 РОЗРОБКА СТАРТАП-ПРОЕКТУ

4.1 Опис ідеї проекту

Ідея проекту (таблиця 4.1) полягає у створенні інтелектуальної системи допомоги вибору житла для покупців та орендаторів за рахунок покращення процесу пошуку найкращої пропозиції з великого об’єму даних за рахунок використання інтелектуальних алгоритмів з одного боку та спрощення процесу надання житла в оренду за допомогою зручного інтерфейсу з іншого.

Призначення системи – це оптимізація та спрощення процесів оренди житла. А саме:

- оптимізація вибору житла для оренди;
- оптимізація формування пропозиції для оренди;
- спрощення процесу укладення угоди про оренду;
- спрощення процесу оплати оренди.

Таблиця 4.1 – Опис ідеї стартап-проекту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Інтелектуальна система з допомоги вибору житла «Homed» з пропозицій для покупців та орендаторів представлена у вигляді веб-сайту.	Надання послуг з покупки та оренди житлових приміщень	Максимальне спрощення та автоматизація процесу вибору найкращої пропозиції житла з численних варіантів.
	Розміщення пропозицій щодо оренди\купівлі.	Можливість фільтрації пропозицій за нестандартними параметрами.

На ринку існують аналоги подібних систем, але більшість з них розробляються лише для вирішення відповідної конкретної задачі, мають однонаправлену конвертацію. Ці аналоги в основному англомовні, дорогі або рідко

оновлюються, а до деяких з них немає доступу без оплати підписки. До того ж розроблена автоматизована програмна система спроектована під цільову аудиторію вітчизняного ринку.

Тому доцільно проводити аналіз потенційних техніко-економічних переваг ідеї порівняно з пропозиціями конкурентів. Результат аналізу у таблиці 4.2 та 4.3.

Таблиця 4.2 – Визначення сильних, слабких та нейтральних характеристик

№ п/п	Техніко- економічні характеристики ідеї
1.	Можливість фільтрації за нестандартними параметрами.
2.	Пришвидшений пошук користувачем ідеальної пропозиції.
3.	Різносторонній аналіз пропозиції.
4.	Використання гнучких алгоритмів, що підвищує продуктивність.
5.	Використання даних геолокації для покращення пошуку.

Таблиця 4.3 – Головні конкуренти

(потенційні) товари/концепції конкурентів				W (слабка сторона)	N (нейтральна сторона)	S (сильна сторона)
Проект	Конкурент 1	Конкурент 2	Конкурент 3			
Інтелектуальна система з допомоги вибору житла	Системи допомоги вибору житла	Системи допомоги вибору житла, з розширеним фільтром	відсутній	Наявність подібних систем.	Використання Google API	Головні конкуренти мають менший функціонал з фільтрації пропозиції.
				Відсутність стартової бази пропозицій.		Використання гнучких алгоритмів, тобто ШІ.

Ідея проекту є актуальною, можна виділити вагомі переваги для споживачів системи. Перелічені техніко-економічні характеристика, слабкі, нейтральні та сильні сторони дають підставу вважати, що проект може мати успіх.

4.2 Технологічний аудит ідеї проекту

Для проведення технічного аудиту ідеї проекту, потрібно провести аудит технологій, за допомогою яких можна реалізувати ідею проекту. І для початку потрібно визначити можливість технологічної здійсненності проекту. Результат представлений у таблиці 4.4.

Таблиця 4.4 – Технологічна здійсненність ідеї проекту

№ п/п	Ідея проекту	Технології її реалізації	Наявність технологій	Доступність технологій
1.	Інтелектуальна система з допомоги вибору житла.	Технологія розробки веб системи, з використанням гнучких алгоритмів.	Технологія наявна	Технологія доступна.

Технологічна реалізація проекту можлива.

Обрана технологія доступна, не потребує доробки, а також безкоштовна та надає усі необхідні можливості для реалізації поставленої задачі. Для розробки з використанням даної технології необхідно мати машину Apple для можливості встановлення робочого середовища.

4.3 Аналіз ринкових можливостей запуску стартап-проекту

4.3.1 Аналіз попиту: наявність попиту, обсяг, динаміка розвитку ринку

Визначення ринкових можливостей, які можна використати під час ринкового впровадження проекту, та ринкових загроз, які можуть перешкодити реалізації проекту, дозволяє спланувати напрями розвитку проекту із урахуванням стану ринкового середовища, потреб потенційних клієнтів та пропозицій проектів-

конкурентів. Для цього спочатку проводиться аналіз попиту (таблиця 4.5).

Таблиця 4.5 – Попередня характеристика потенційного ринку стартап-проекту

№ п/п	Показники стану ринку (найменування)	Характеристика
1	Кількість головних гравців, од	Автори алгоритмів, технічна підтримка, розробники, користувачі
2	Загальний обсяг продаж, грн/ум.од	270 тыс.грн.
3	Динаміка ринку (якісна оцінка)	Зростає
4	Наявність обмежень для входу (вказати характер обмежень)	Наявність конкурентів
5	Специфічні вимоги до стандартизації та сертифікації	Система повинна відповідати вимогам, які диктують системи проведення платежів
6	Середня норма рентабельності в галузі (або по ринку), %	14%

Висновок: Проект є привабливим для входу на ринок.

Надалі визначаються потенційні групи клієнтів, їх характеристики, та формується орієнтовний перелік вимог до товару для кожної групи (таблиця 4.6). Після визначення потенційних груп клієнтів проводиться аналіз ринкового середовища: складаються таблиці факторів, що сприяють ринковому впровадженню проекту, та факторів, що йому перешкоджають.

Результати представлені у таблицях 4.7 та 4.8 відповідно.

Після аналізу конкуренції проводиться більш детальний аналіз умов конкуренції в галузі (табл. 4.10) -за моделлю п’яти сил М. Портера, яка вирізняє п’ять основних факторів, що впливають на привабливість вибору ринку з огляду на

характер конкуренції:

- конкурент, що вже є у галузі;
- потенційні конкуренти;
- наявність товарів-замінників;
- постачальники, що конкурують за ринкову владу;
- споживачі, які конкурують за ринкову владу.

4.3.2 Визначення груп потенційних клієнтів

Таблиця 4.6 – Характеристика потенційних клієнтів стартап-проекту

№ п/п	Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
	Потреба у зручному житлі.	Орендодавці	Орендодавці націлені на прибуток з оренди	Прийнятна цінова політика. Висока ефективність пошуку клієнтів
		Орендарі	Орендарі націлені на зручне житло.	Більша ефективність пошуку пропозицій порівняно з аналогічними системами.
		Продавці житла	Продавці житла націлені на прибуток з продажу	Висока надійність клієнтів

Продовження таблиці 4.6

				покупців та гарантії на юридично правильне оформлення
		Покупці житла	Покупці житла націлені на зручне житло.	Висока якість пропозицій та юридичний захист

Визначена характеристика дозволяє зробити висновок, що проект знайде свого покупця.

4.3.3 Аналіз ринкового середовища

Таблиця 4.7 – Фактори загроз

№ п/п	Фактор	Зміст загрози	Можлива реакція компанії
1.	Недостатня розвиненість інвестиційної діяльності у країні.	Коштів може бути не достатньо для реалізації задуманої ідеї.	Пошук компаній, які готові бути партнерами, волонтерів, вітчизняних розробників. Реклама партнерської продукції.
2.	Зростання вимог споживачі	З впровадженням системи вимоги до неї можуть змінюватись.	Реалізація оновлень до систем.

Таблиця 4.8 – Фактори можливостей

№ п/п	Фактор	Зміст можливості	Можлива реакція компанії
1.	Зацікавленість інвесторів	Інвестори можуть бути зацікавленні у	Зацікавленість інвесторів може допомогти залучити експертів

Продовження таблиці 4.8

		нестандартному підході реалізації та вирішенні проблеми пошуку житла.	для розробки ефективного і надійного ПЗ.
2.	Новизна ідеї	Незвичайна ідея може залучити нових партнерів.	Налагодження зв’язків, залучення нових партнерів.
4.	Нові проекти	На базі запропонованої ідеї можна розробити додаткові проекти та системи вибору житла.	Активна роль у розвитку нових гілок справи, які можу збільшити прибуток.

Висновок: стартап проект можна впроваджувати на ринок.

Надалі проводиться аналіз пропозиції – визначаються загальні риси конкуренції на ринку (таблиця 4.9): визначаються тип можливої майбутньої конкуренції та її інтенсивність, рівень конкурентоспроможності за рівнем конкурентної боротьби, видами товарів і галузевою ознакою.

4.3.4 Аналіз пропозиції

Таблиця 4.9 – Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	В чому проявляється дана характеристика
1. Вказати тип конкуренції - монополія/олігополія/ монополістична/чиста	Чиста
2. За рівнем конкурентної боротьби - локальний/національний/…	Локальний
3. За галузевою ознакою - міжгалузева/ внутрішньогалузева	внутрішньогалузева
4. Конкуренція за видами товарів: - товарно-родова - товарно-видова	Товарно-родова
5. За характером конкурентних переваг - цінова / нецінова	нецінова
6. За інтенсивністю - марочна/немарочна	немарочна

Висновок: конкуренція для реалізації проекту на ринку прийнятна.

4.3.5 Аналіз умов конкуренції в галузі 5 сил М. Портера

Таблиця 4.10 – Аналіз конкуренції в галузі за М. Портером

Складові аналізу	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товари-замінники
	Аналогічні системи на ринку.	Нові системи на ринку	Розробники бібліотек, фреймворків та алгоритмів, які можна використовувати у розробках.	Можливість гнучкої фільтрації пропозиції за нестандартними параметрами.	Аналогічні системи
Висновки:	Так як розроблена система має суттєві функціональні переваги, то інтенсивність щодо боротьби припустима.	Враховуючи, що при впровадженні таких систем не має стартових пропозицій, то бар’єр для входу на ринок прийнятний.	Постачальники не диктують умови на ринку, а лише надають інструмент для реалізації.	Головними умовами є вирішення поставленої ним задачі. Клієнти диктують умови на ринку.	Обмеження на ринку з боку товарів замінників – мінімальна.

Висновок: проект можу бути впроваджений на ринку з огляду на конкурентну ситуацію.

4.3.6 Перелік факторів конкурентоспроможності

Перелік факторів конкурентоспроможності

- гнучкість використання: система може бути адаптована на нові вимоги ринку;
- продуктивність: запропонована система використовує нейрону мережу, що підвищує ефективність допомоги вибору житла;
- новизна – запропонована система використовує еволюційний алгоритм навчання.

Висновок: зазначені фактори надають проекту можливість виходу на ринок, так попиту споживачів.

4.3.7 Аналіз сильних та слабких сторін стартап-проекту

Сильні та слабкі сторони проекту описані в таблиці 4.11.

Таблиця 4.11 – Порівняльний аналіз сильних та слабких сторін «назва проекту»

№ п/п	Фактор конкурентоспроможності	Бали	Прямі конкуренти	Система «Homed»
1	Гнучкість використання	1-20	5	15
2	Продуктивність	1-20	15	18
3	Новизна	1-20	10	20
			30	53

Висновок: Система «Homed» вище завдяки ціновому фактору та новизні запропонованого рішення.

4.3.8 SWOT-аналіз

Фінальним етапом ринкового аналізу можливостей впровадження проекту є складання SWOT-аналізу (Strength, Weak, Opportunities, Troubles) (таблиця 4.12) на основі виділених ринкових загроз та можливостей, та сильних і слабких сторін.

Таблиця 4.12 – SWOT- аналіз стартап-проекту

Сильні сторони стартап-проекту	Слабкі сторони стартап-проекту
Маркетинг	
Налагодження партнерських зв'язків, просування продукту на вітчизняному ринку.	Недостатній імідж на ринку.
Розробка	
Застосування новітніх технологій при розробці.	Складність гнучких алгоритмів
Персонал	
Ефективна кадрова політика, професійність та кваліфікованість кадрів, залучення українських розробників та міжнародних експертів.	Відсутність достатнього досвіду.
Дослідження та розробки	
Постійне оновлення продукції, дослідження направлені на покращення якості системи.	Можливі додаткові витрати на навчання персоналу при розробках.
Фінанси	
Основна сума статутного капіталу формується за рахунок власних фінансових ресурсів учасників, вкладень інвесторів, замовників.	Можливі додаткові фінансові витрати при зміні конфігурацій, оснащення та купівлі обладнання.
Можливості	Загрози

Продовження таблиці 4.12

Прихильність до впровадження нових технологій на ринок.	Зміна політики непрямих конкурентів.
Розширення функціоналу системи	Нестабільна політична та економічна ситуація.
Послаблення позицій конкурентів.	Наявність прямих конкурентів.
Використання новітніх світових технологій та засобів розробки.	Технологічна невідповідність
Залучення висококваліфікованого персоналу.	Не прийняття новизни та запропонованого рішення проекту.
Задоволення запитів споживача: час вибору пропозиції, швидкість пошуку, якість оцінки пропозиції.	
Збільшення прихильності клієнтів, за рахунок налагодженню партнерських зв'язків.	

Висновок: проведений SWOT-аналіз показав, що стартап-проект доцільно реалізовувати.

На основі SWOT-аналізу розробляються альтернативи ринкової поведінки (перелік заходів) для виведення стартап-проекту на ринок та орієнтовний оптимальний час їх ринкової реалізації з огляду на потенційні проекти конкурентів, що можуть бути виведені на ринок. Визначені альтернативи аналізуються з точки зору строків та ймовірності отримання ресурсів (таблиця 4.13).

4.3.9 Альтернативи ринкової поведінки

Таблиця 4.13 Альтернативи ринкової поведінки

Альтернатива ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
Запровадження у систему нових технологій кластеризації, класифікації, фільтрації та нових гнучких алгоритмів	Розширення клієнтської бази	Від 6 до 12 місяців
Використання висококваліфікованого персоналу	Підвищення якості продукту та швидкість розробки	Від 2 місяців до 6
Формування позитивного іміджу при задоволенні зростаючого попиту	Розширення клієнтської бази	Від 2 місяців до 6
Орієнтація на різні вподобання користувачів, їх вподобання щодо ідеального житла	Розширення клієнтської бази	Від 6 до 9 місяців
Вихід на нові ринки	Пошук інвесторів Розширення клієнтської бази	Від 1 до 4 місяців

4.3.10 Висновки до аналізу ринкових можливостей запуску стартап-проекту

Є можливість ринкової комерціалізації проекту, наявний попит та рентабельність роботи на ринку;

Є перспективи впровадження з огляду на потенційні групи клієнтів, бар’єри входження, стан конкуренції, конкурентоспроможність проекту;

Доцільною є подальша імплементація проекту.

4.4 Розроблення ринкової стратегії

Розроблення ринкової стратегії першим кроком передбачає визначення стратегії охоплення ринку: опис цільових груп потенційних споживачів, які визначені у таблиці 4.14.

Таблиця 4.14 – Вибір цільових груп потенційних споживачів

Опис цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в сегменті	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
Орендатори	Потребують	Попит є	Присутня	Помірно
Продавці	Потребують	Попит є, проте нижчий ніж у орендаторів	Присутня	Помірно
Орендарі	Потребують	Попит є	Присутня	Помірно
Покупці	Потребують	Попит є, проте нижчий ніж у орендарів	Присутня	Помірно

Які цільові групи обрано: оскільки різниця між цільовими групами зовсім незначна, а також враховуючи той факт, що компанія має бажання почати продажі (а відповідно і отримання прибутку) якомога швидше, то доцільно враховувати усі цільові групи, тобто використовувати масовий маркетинг, пропонуючи стандартизовану програму.

За результатами аналізу потенційних груп споживачів (сегментів) автори ідеї обирають цільові групи, для яких вони пропонуватимуть свій товар, та визначають стратегію охоплення ринку.

Для роботи в обраних сегментах ринку необхідно сформувати базову стратегію розвитку, яка визначається у таблиці 4.15. Вибір стратегії конкурентної поведінки визначається у таблиці 4.16.

Таблиця 4.15 – Визначення базової стратегії розвитку

Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку
Вихід на нові ринки	Стратегія спеціалізації	Надання товару із варіативністю локалізації	Стратегія диференціації
Розширення клієнтоорієнтованого функціоналу	Стратегія диференціації (допускається стратегія спеціалізації)	Надання товару відмінних якостей, які роблять систему особливою на фоні аналогічних розробок	Стратегія диференціації (допускається стратегія спеціалізації)

Таблиця 4.16 – Визначення базової конкурентної поведінки

Чи є проект «першопроходцем» на ринку	Так, на локальному.
Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Обидва варіанти
Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Ні
Стратегія конкурентної поведінки	Стратегія виклику лідера

На основі вимог споживачів з обраних сегментів до постачальника (стартап-компанії) та до продукту, а також в залежності від обраної базової стратегії розвитку

та стратегії конкурентної поведінки розробляється стратегія позиціонування (таблиця 4.17), що полягає у формуванні ринкової позиції (комплексу асоціацій), за яким споживачі мають ідентифікувати торгівельну марку або проект.

Таблиця 4.17 – Визначення стратегії позиціонування

Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту
Доступна ціна, простота і зручність використання, універсальність	Стратегія диференціації	Вирішення важливих поставлених задач швидко, легко та зрозуміло навіть без інструкцій. Легкість і простота у використанні. Доступність через ціну та технічні характеристики	– стандарти якості – метрики ПЗ – ASQAS - automated system of quality assessment software

Результатом є узгоджена система рішень щодо ринкової поведінки стартапкомпанії, яка визначатиме напрями роботи стартап-компанії на ринку. Отже, робота стартап-компанії на ринку повинна бути спланована орієнтовано таким чином: за стратегією диференціації виконаний і буде поширюватись товар відмінний за

властивостями від своїх аналогів, дотримуючись у конкурентній поведінці стратегії «виклику лідера», тобто випускається один товар для усіх можливих споживачів.

Надалі розроблена трирівнева маркетингова модель товару: уточняються ідея продукту, його фізичні складові, особливості процесу його надання.

4.5 Розробка маркетингової програми стартап-проекту

Першим кроком є формування маркетингової концепції товару, який отримає споживач. Для цього у таблиці 4.18 підсумовані результати попереднього аналізу конкурентоспроможності товару, а в таблиці 4.19 представлено опис трьох рівнів моделі товару.

Таблиця 4.18 – Визначення ключових переваг концепції потенційного товару

Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами
Оцінка якості ПЖ	Оцінка за 4 метриками. Удосконалення оцінки будь-якої з обраних характеристик.	Розрахункові показники, точність та достовірність яких можна оцінювати; кількість вхідних параметрів; самостійність програмної системи.

Таблиця 4.19 – Опис трьох рівнів моделі товару

Рівні товару	Сутність та складові
Товар за задумом	Перегляд списку пропозицій Фільтрація за стандартними параметрами Формування вибірки пропозицій за гнучкими та інтелектуальними критеріями Створення та редагування облікових записів Рейтингова система
Реалізований товар	Реалізовано перегляд списку пропозицій Реалізовано фільтрацію за стандартними параметрами Реалізовано формування вибірки пропозицій за гнучкими та інтелектуальними критеріями Реалізовано створення та редагування облікових записів Реалізовано рейтингову систему

За рахунок чого потенційний товар буде захищено від копіювання: від копіювання потенційний товар захистити не складає проблеми. Розроблена математична модель підбору пропозицій, на якій базується програмна система, публікувалась лише у загальних рисах, а без математичної моделі цей ПП лише набір рядків коду. Визначення цінових меж, якими необхідно керуватись при встановленні ціни на потенційний товар, яке передбачає аналіз ціни на товари-

аналоги, а також аналіз рівня доходів цільової групи споживачів описано в таблиці 4.20.

Таблиця 4.20 – Визначення меж встановлення ціни

Рівень цін на товарианалоги (середнє за місяць)	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни
10000 грн	40000 грн	4000-30000 грн

Наступним кроком є визначення оптимальної системи збуту, в межах якого приймається рішення (таблиця 4.21): проводити збут власними силами або залучати сторонніх посередників, вибір та обґрунтування оптимальної глибини каналу збуту, вибір та обґрунтування виду посередників.

Таблиця 4.21 – Формування системи збуту

Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
Бажання отримати більше за менші гроші	Залучення клієнтської бази та продаж	Нульовий рівень: тільки виробник	Вертикальна маркетингова система

Останньою складової маркетингової програми є розроблення концепції маркетингових комунікацій, що спирається на попередньо обрану основу для позиціонування, визначену специфіку поведінки клієнтів (таблиця 4.22).

Таблиця 4.22 – Формування системи збуту

Поведінка цільових клієнтів	Канали комунікацій цільових клієнтів	Ключові позиції, обрані для позиціонування	Завдання рекламного повідомлення
Бажання отримати більше за менші гроші	Будь-які, але бажано з великою кількістю візуального контенту	Низька ціна Широкий вибір функціоналу Легкий і простий у використанні продукт	Донести до користувача суть продукту, його якість, та залучити якомога більше зацікавлених клієнтів

4.6 Економічне обгрунтування розробки

У даному підрозділі наведено економічне обгрунтування створення даного проекту: загальні витрати на розробку, супровідні витрати, можливі джерела інвестицій, строки та умови повернення займаних коштів. З метою реалізації проекту організовується ФОП, форма оподаткування ЄСВ, ставка 6%, об'єкт оподаткування - «доходи мінус витрати». Процес реалізації можна розбити на два етапи: підготовчий і операційний. До підготовчого відноситься підбір персоналу, організація робочих місць, проплата ліцензій на програмне забезпечення.

Таблиця 4.23 – Формування системи збуту

Позиція	Необхідний строк задіяності на проекті	Заробітна плата($\міс)
Архітектор\	8 місяців	2000

Продовження таблиці 4.23

Проектний менеджер		
Розробник клієнтської частини	6 місяців	1400
Розробник серверної частини	6 місяців	1800
Тестувальник	4 місяці	350
Маркетолог	8 місяців	500
Всього:	8 місяців	40 600

Із таблиці 4.23 бачимо, що загальний час на розробку – 8 місяців, з яких 2 перших місяці займатиме планування архітектури та розробка маркетинг-плану продукту. Із 3го місяця починатиметься розробка програмного продукту. Із 5го місяця запланований початок тестування. Далі наведена таблиця із супровідними витратами, необхідними на час розробки проекту.

Таблиця 4.24 – Супровідні витрати

Позиція	Необхідний строк задіяності на проекті	Вартість
Оренда приміщення	8 місяців	3200
Оренда офісних меблів	8 місяців	1600
Технічне забезпечення	-	8000
Програмне забезпечення	8 місяців	2000

Продовження таблиці 4.24

Офісне приладдя	-	300
Логістика	8 місяців	600
Реклама та просування	-	12 000
Форс-мажорні витрати	8 місяців	2000
Всього:	8 місяців	29 700

Маючи інформацію про необхідні матеріальні та нематеріальні активи, а також про очікувані річні обсяги збуту створеного продукту із таблиці 4.24, отримуємо наступні дані Обсяг інвестиційних витрат – 70 300 у.о., з яких власні кошти ініціатора проекту – 10 000 у.о.. Нестачу коштів планується покрити за допомогою залучення банківського кредиту на термін 48 місяців зі ставкою 14% річних. Кредитні канікули три місяці. Виплата здійснюється ануїтетними платежами.

4.7 Висновки до розділу

Отже, ринкова (маркетингова) програма орієнтовано має бути побудована таким чином:

- розробка продукту;
- вибір сегменту ринку та пошук клієнтів;
- стратегія розвитку - стратегія розподіленості, тобто формування конкурентоспроможності досягається шляхом надання споживачу товару, якого той потребує. На основі детального вивчення середовища споживання розробляється одна або декілька особливих характеристик власного товару;

– стратегія конкурентної поведінки – стратегія виклику лідера, тобто на споживчому ринку націлюватись на всіх можливих споживачів, у тому числі клієнтів фірм-конкурентів. Така стратегія будується за принципом «йти слідом» за лідером ринку. За наступні цілі ставиться можливість обійти лідерів цільового сегменту.

Стан та динаміка ринкового середовища на сьогодні і ще багато років є і будуть залишатись сприятливими для впровадження розробленої системи, а також для її необхідності.

Конкурентні переваги створеного продукту очевидні. На вітчизняному ринку аналогів майже не існує, а існуючі – вкрай низької якості. На міжнародному ринку конкуренція наявна та буде рости, якщо не підтримувати та не розвивати свій продукт.

Також, після проведення аналізів можливого цільового сегменту (споживачів), потреб споживачів та можливого попиту, динаміки ринку та рентабельності роботи на ринку, можна однозначно зробити висновок, що створений проект доцільний до комерціалізації.

Перспективи впровадження з огляду на потенційні групи клієнтів, бар’єри входження, стан конкуренції та конкурентоспроможності проекту – прямі, і тільки доводять можливість впровадження, та не марну розробку створеного продукту.

ВИСНОВКИ

Визначено основні проблеми у існуючих системах допомоги вибору житла з пропозицій оренди для орендарів, серед яких є відсутність інтелектуального пошуку та фільтрів пропозицій. Для подолання цієї проблеми розроблено інтелектуальний алгоритм попередньої обробки пропозицій оренди. Сформовано функціонал системи, визначені бізнес-процеси та ролі.

Досліджено архітектуру та технології розробки для побудови системи допомоги вибору житла. У результаті використано: Visual Studio Code; PostgreSQL, FireBase – реалізація автентифікації; Node.js, AdonisJs, ReactJS – для програмної реалізації та інтерфейсу користувача, мову Python – для розробки методів кластеризації та класифікації. Використання цих сучасних технологій дозволяє досягти високих показників ефективності системи за мінімальну вартість розробки.

Створено три мікро сервіси:

– веб сервер для обробки запитів користувачів;
– сервіс кластеризації;
– сервіс класифікації.

Використання мікро сервісної архітектури дає можливість масштабувати та редагувати кожен мікро сервіс окремо, що значно спрощує задачу удосконалення та масштабування системи.

Результатом магістерської роботи стала інтелектуальна система допомоги вибору житла з пропозицій оренди для орендарів. Система оснащена зручним користувацьким інтерфейсом та задовольняє усі поставлені вимоги до функціональності системи.

ПЕРЕЛІК ПОСИЛАНЬ

1) Райзин Д. В. Классификация и кластеризация. – М.:Мир, 1980. – 390 с.

2) L. Kaufman. Finding Groups in Data: An Introduction to Cluster Analysis. / L. Kaufman, P. J. Rousseeuw. – MA: MIT Press, 2007 – С. 38-49.

3) Мандель И. Д. Кластерный анализ. – М.: Финансы и статистика, 1988. – С. 96-101.

4) Jain A., Murty M., Flynn P. 1999. Data Clustering: A Review, volume 31. ACM Computing Surveys.

5) Sudipto Guha, Rajeev Rastogi, Kyuseok Shim «CURE: An Efficient Clustering Algorithm for Large Databases». Электронное издание.

6) Jure Lescovec, Anand Rajaraman, Jeffrey D. Ullman. Mining of Massive Datasets. – 2014. – С. 254-262.

7) Болдак, А.А.; Сухарев, Д.Л., «Определение количества кластеров в статистических данных,» Вісник НТУУ «КПІ». Інформатика, управління та обчислювальна техніка, 2011. – С. 118–122.

8) Дубров А. М. Многомерные статистические методы: Учебник. / Дубров А. М., Мхитарян В. С., Трошин Л. И. – К.: Финансы и статистика, 2000. – С. 5-7.

9) Барсегян А.А., Куприянов М.С. и др. Методы и модели анализа данных: OLAP и Data Mining. - СПб.: БХВ , 2004. – 160 с.

10) Pal N. R. et al. A possibilistic fuzzy c-means clustering algorithm //Fuzzy Systems, IEEE Transactions on. 2005. – С. 517-530.

11) A. K. Jain. Algorithms for Clustering Data. / A. K. Jain., R. C. Dubes – Prentice Hall, IGI Global, 2012 – С. 143-243.

Додаток А

Інструкція користувача

На початку роботи із системою Homed, користувач потрапляє на головну сторінку (рисунок Б.1).

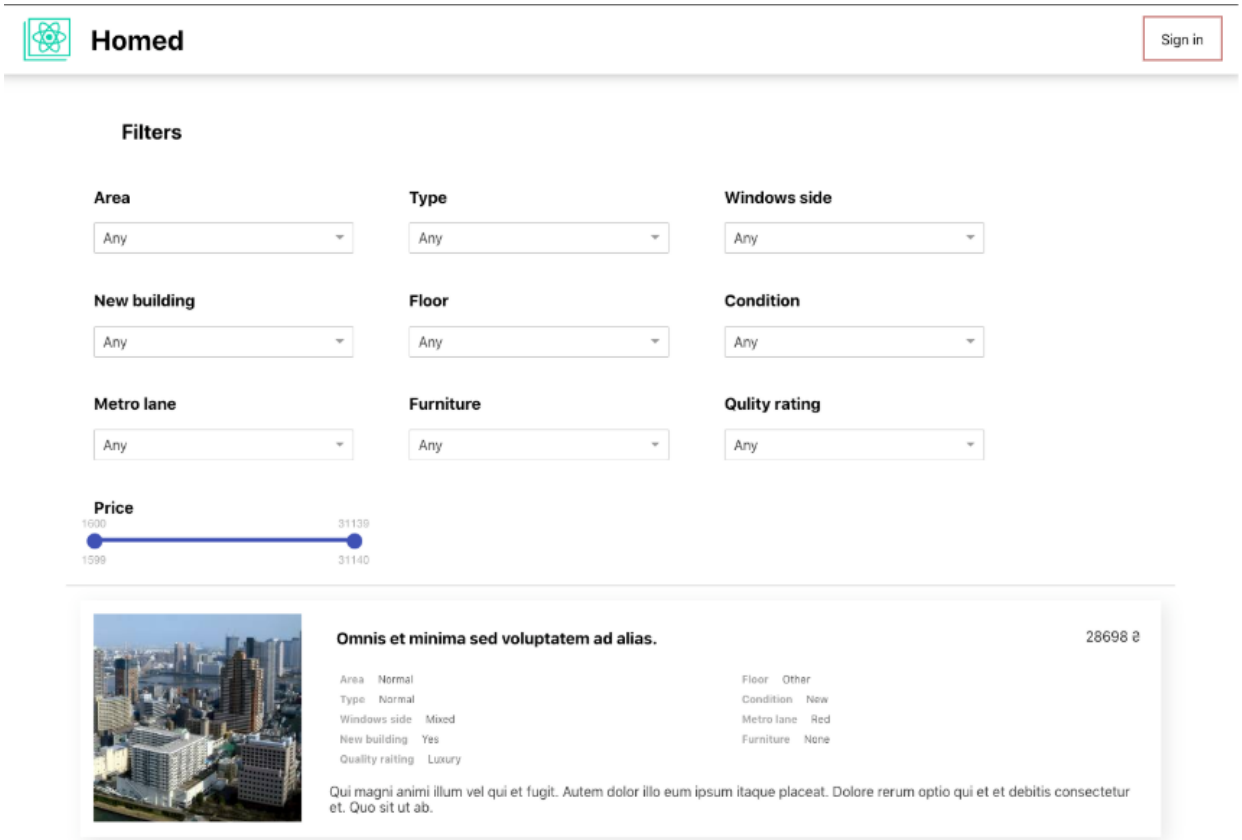


Рисунок Б.1 – Головна сторінка Homed

На головній сторінці наявний набір фільтрів, представлений у таблиці Б.1.

Таблиця Б.1 – Опис фільтрів

Фільтри	Опис
Area	Фільтр Area – площа житла. У даному фільтрі користувач може обрати площу бажаного житла.
Type	Фільтр Type – тип житла. У даному фільтрі можна обрати тип квартири за плануванням від квартири студії, до двоповерхової квартири.
Windows side	Фільтр Windows side – сторона, на яку виходять вікна. За допомогою даного фільтру користувач може обрати квартиру зі сонячною або темною стороною.

Продовження таблиці Б.1

New building	Фільтр New Buildings – нова будівля. За допомогою даного фільтру користувач може обрати нову будівлю чи будівлю яка вже давно в експлуатації.
Floor	Фільтр Floor – поверх. За допомогою даного фільтру користувач може обрати бажаний поверх у житлі.
Condition	Фільтр Condition – стан житла. За допомогою даного фільтру користувач може обрати стан квартири, чи необхідно в ній роботи ремонт, чи квартира є новою.
Metro lane	Фільтр Metro line – лінія метро. За допомогою цього фільтру користувач може обрати гілку метро біля якого хоче знайти житло.

Продовження таблиці Б.1

Furiture	Фільтр Furniture – меблі. За допомогою даного фільтру можна визначити стан меблів у житлі.
Quality rating	Фільтр Quality rating – рейтинг якості. За допомогою даного рейтингу можна оцінити якість житла, яке надається до оренди.

За замовчуванням у кожному фільтрі стоїть позначка «Any» - означає, що дана характеристика може бути будь-якою.

Нижче розташований бігунок (рисунок Б.2) за допомогою якого можна обрати діапазон цін, що цікавлять користувача, сума наведена за оплату житла за місяць.

Рисунок Б.2 – Вибір ціни

Далі представлений перелік пропозицій житла (рисунок Б.3) за заданими фільтрами.

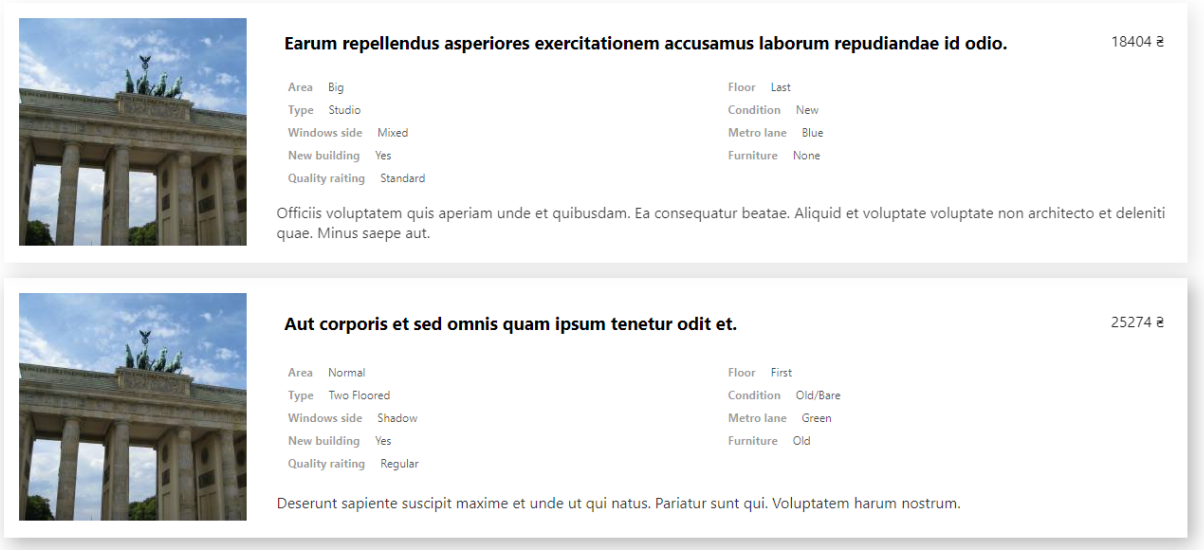


Рисунок Б.3 – Список пропозицій

Перелік представлений нескінченним списком. Для того, щоб піднятися вгору на головну сторінку необхідно натиснути кнопку «Go up» (рисунок Б.4).

Рисунок Б.4 – Кнопка «Go up»

Незареєстрований користувач обмежений у функціоналі, тому кожен відвідувач сайту може зареєструватися або авторизуватися (рисунок Б.5).

Рисунок Б.5 – Кнопка реєстрації

Після натиснення кнопки користувач попадає на сторінку із вводом логіну та паролю для реєстрації або авторизації (рисунок Б.6).

Рисунок Б.6 – Форма реєстрації

Після реєстрації, зареєстрований користувач попадає на головну сторінку. Проте, йому тепер доступний повний функціонал.

Зареєстрований користувач може створювати власну пропозицію житла та виставляти її на сайт. Для цього необхідно у верхньому правому куті натиснути кнопку для додання нової пропозиції (рисунок Б.7).

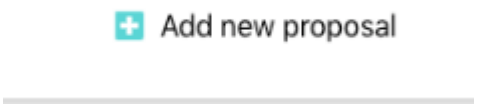


Рисунок Б.7 – Кнопка додання нової пропозиції

Після натиснення кнопки користувач попадає на сторінку додавання пропозиції зображену на рисунку Б.8.

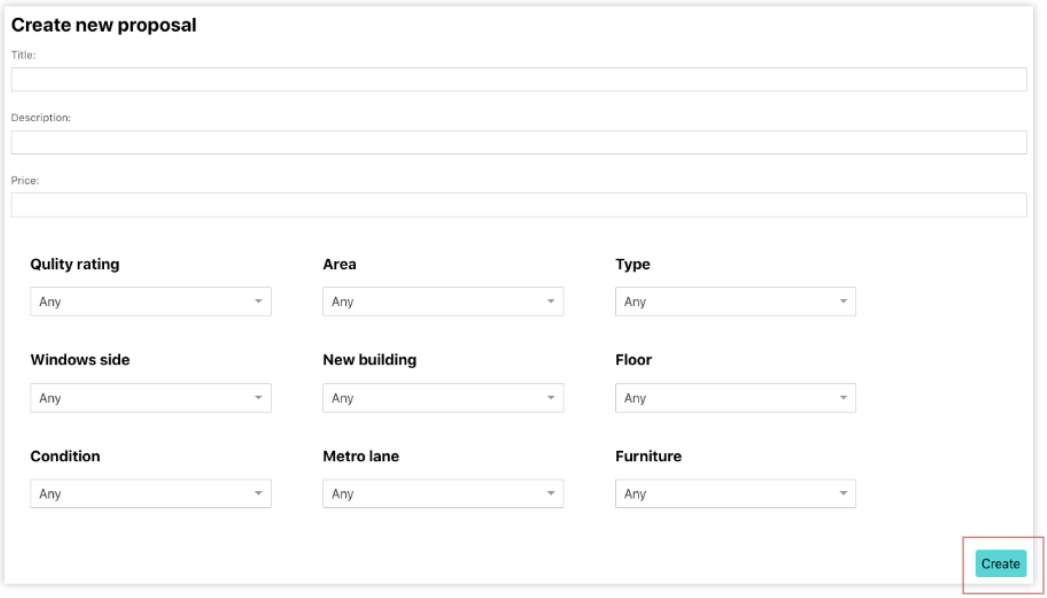


Рисунок Б.8 – Сторінка створення пропозиції

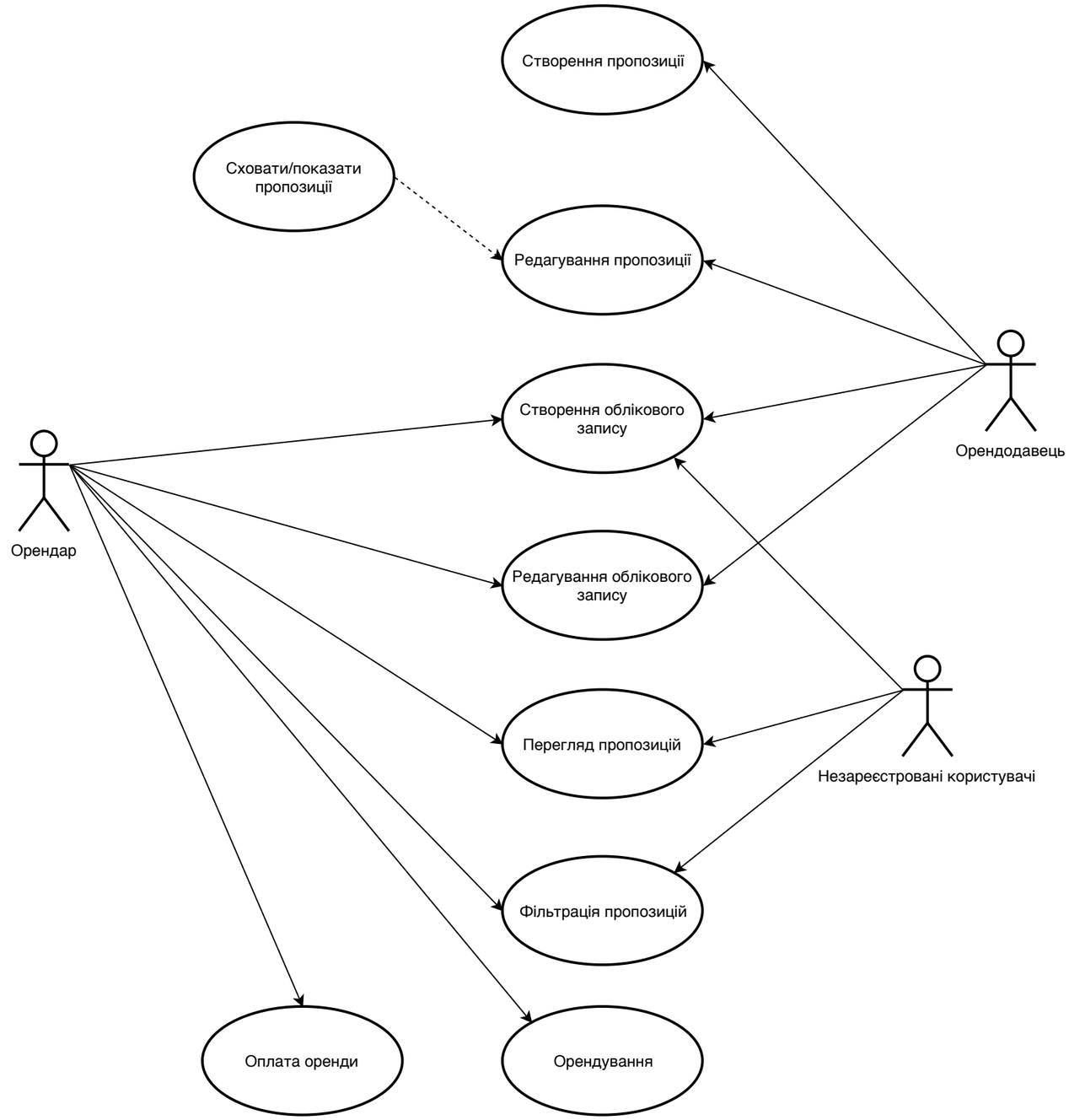
На сторінці можна ввести параметри квартири, які були описані вище, як фільтри, додати опис (те що не входить у визначені параметр або додаткову інформацію) та надати ціну пропозиції.

Після створення пропозиції користувач потрапляє на головну сторінку.

Додаток Б

Графічний матеріал

Діаграма прецедентів



Демонстраційний плакат до магістерської дисертації

«*Інтелектуальна система підтримки вибору житла для покупців та орендаторів*»

Магістрант Меліков Євгеній Олексійович

Керівник Тєлишева Тама Олексіївна

Математична модель

Алгоритм кластерізації

$$e^2 = \sum_{j=1}^{K} \sum_{i=1}^{n_j} \left\| x_{ij} - c_j \right\|^2$$

де вхідними даними є:

x – елемент кластеру, тобто параметр пропозиції

оренди в даному випадку;

K – кількість кластерів;

c – централи кластерів.
вихідні дані:

e – квадратична помилка розбиття.

Оціночна функція

$$q(P, C) = \sum_{i=1}^{n} P_i C_i$$

$$q_r(P, C) = \frac{q(P, C)}{P_{\text{ціна}}}$$

$$d(x, x') = \sqrt{\sum_{i=1}^{n} (x_i - x_i')^2}$$

$$p(P, P', C) = \sqrt{\sum_{i=1}^{n} \frac{(P_i - P'_i)^2 {C_i}^2}{\left(P_{\text{ціна}}^{(i)} - P'^{(i)}_{\text{ціна}}\right)^2}}$$

де вхідними даними є:

P– вектор нормованих параметрів пропозицій;

C – вагові коефіцієнти параметрів;

x та x' - це елементи n вимірного простору, що описують дві точки між якими визначається відстань;

P, P' - цепорівнювані пропозиції оренди, що представлені векторами параметрів.

вихідні дані:

q - кількісна оцінка якості пропозиції оренди;

d - евклідова відстань між точками у n вимірному просторі;

q_p - відношення оцінки якості до ціни пропозиції;

p - оцінка відстані між пропозиціями за якістю.

Алгоритм класифікації

$$E^2(X, U) = \sum_{i=1}^{N} \sum_{k=1}^{K} U_{ik} \left\| x_i^{(k)} - c_k \right\|^2$$

де вхідними даними є:

X – матриця даних пропозицій;

U – матриця відношень пропозицій до кластерів;

K– кількість кластерів;

N– кількість параметрівмоделі пропозиції оренди

вихідні дані:

E – матриця помилки розбиття.

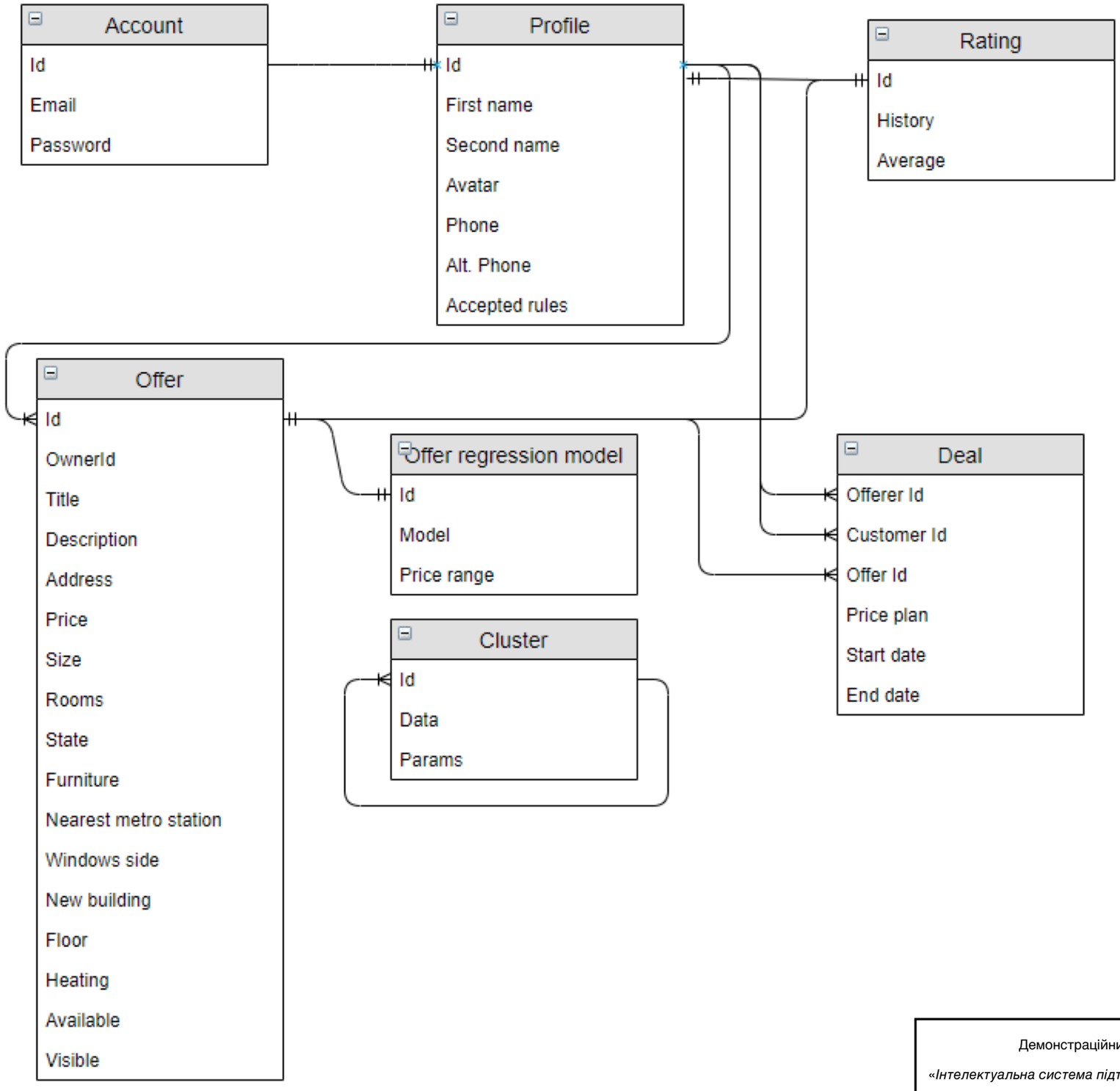
Демонстраційний плакат до магістерської дисертації

«*Інтелектуальна система підтримки вибору житла для покупців та орендаторів*»

Магістрант Меліков Євгеній Олексійович

Керівник Тєлишева Тама Олексіївна

Концептуальна схема бази даних



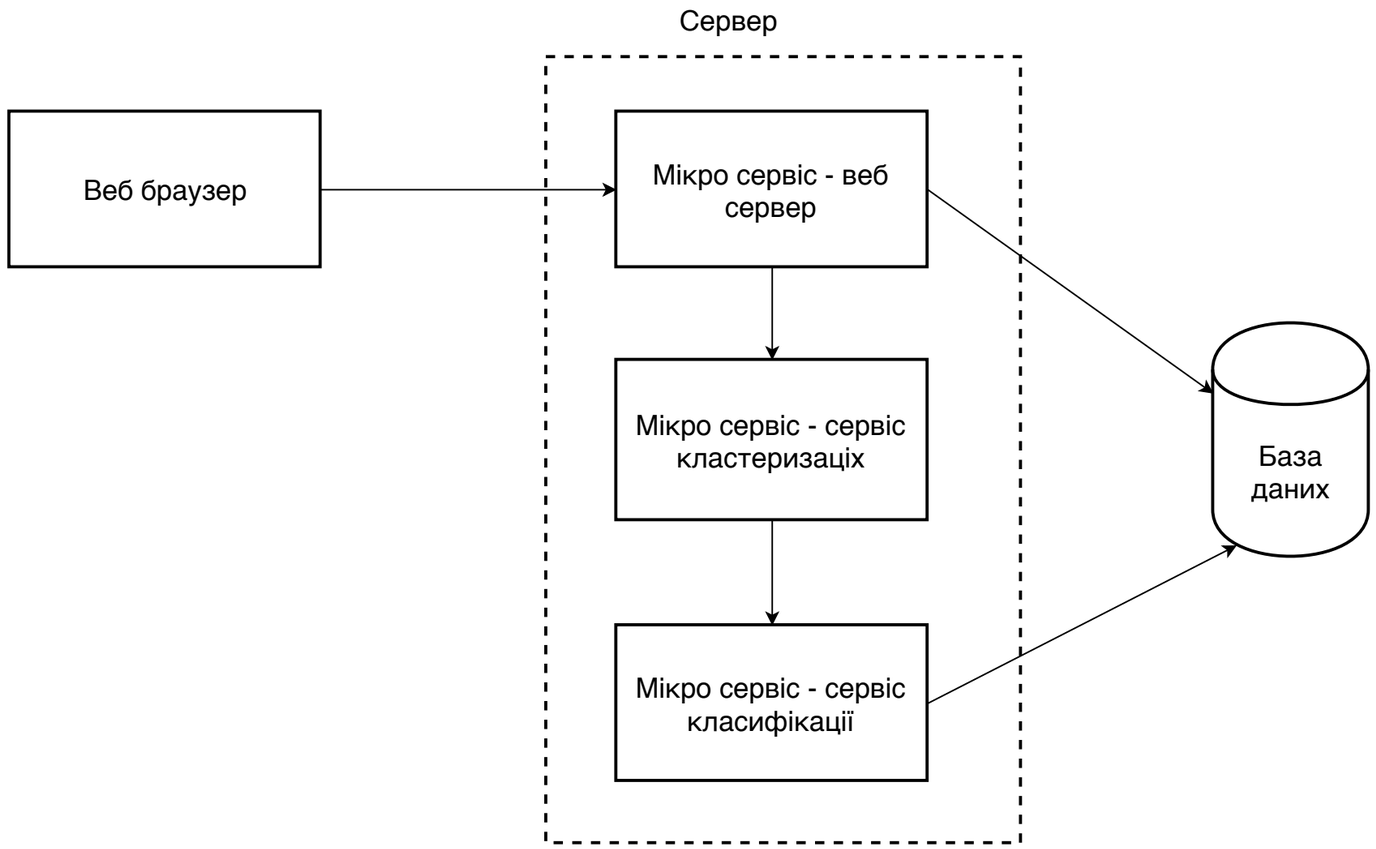
Демонстраційний плакат до магістерської дисертації

«*Інтелектуальна система підтримки вибору житла для покупців та орендаторів*»

Магістрант Меліков Євгеній Олексійович

Керівник Тєлишева Тама Олексіївна

Архітектура системи



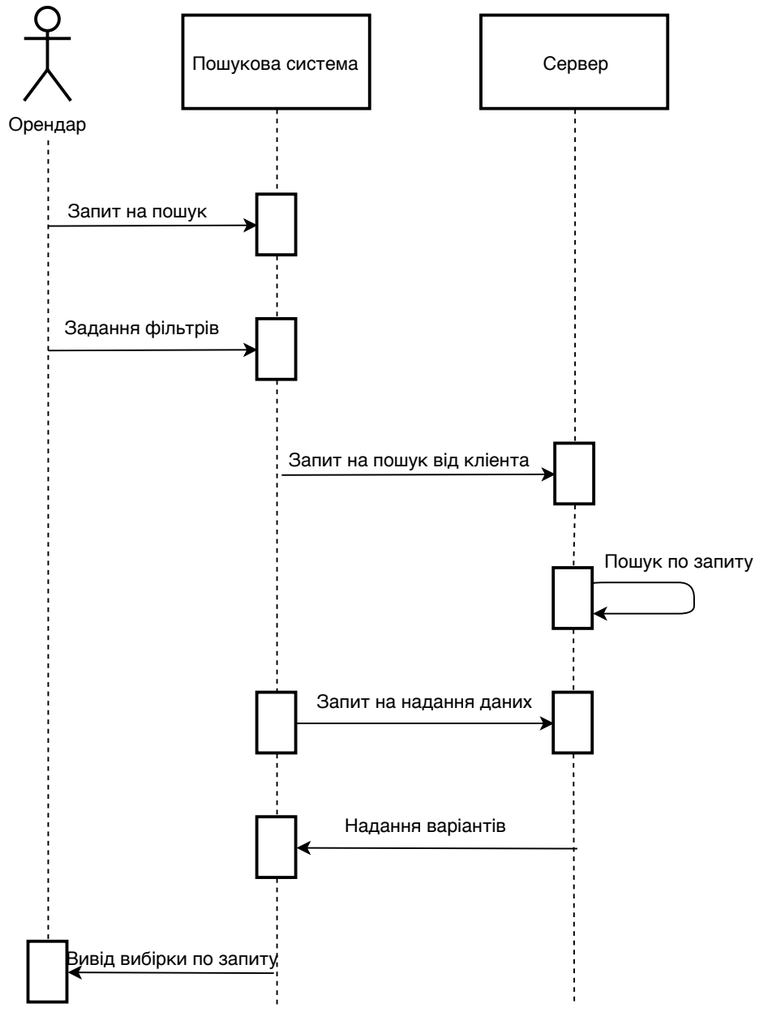
Демонстраційний плакат до магістерської дисертації

«*Інтелектуальна система підтримки вибору житла для покупців та орендаторів*»

Магістрант Меліков Євгеній Олексійович

Керівник Тєлишева Тама Олексіївна

Діаграма послідовності



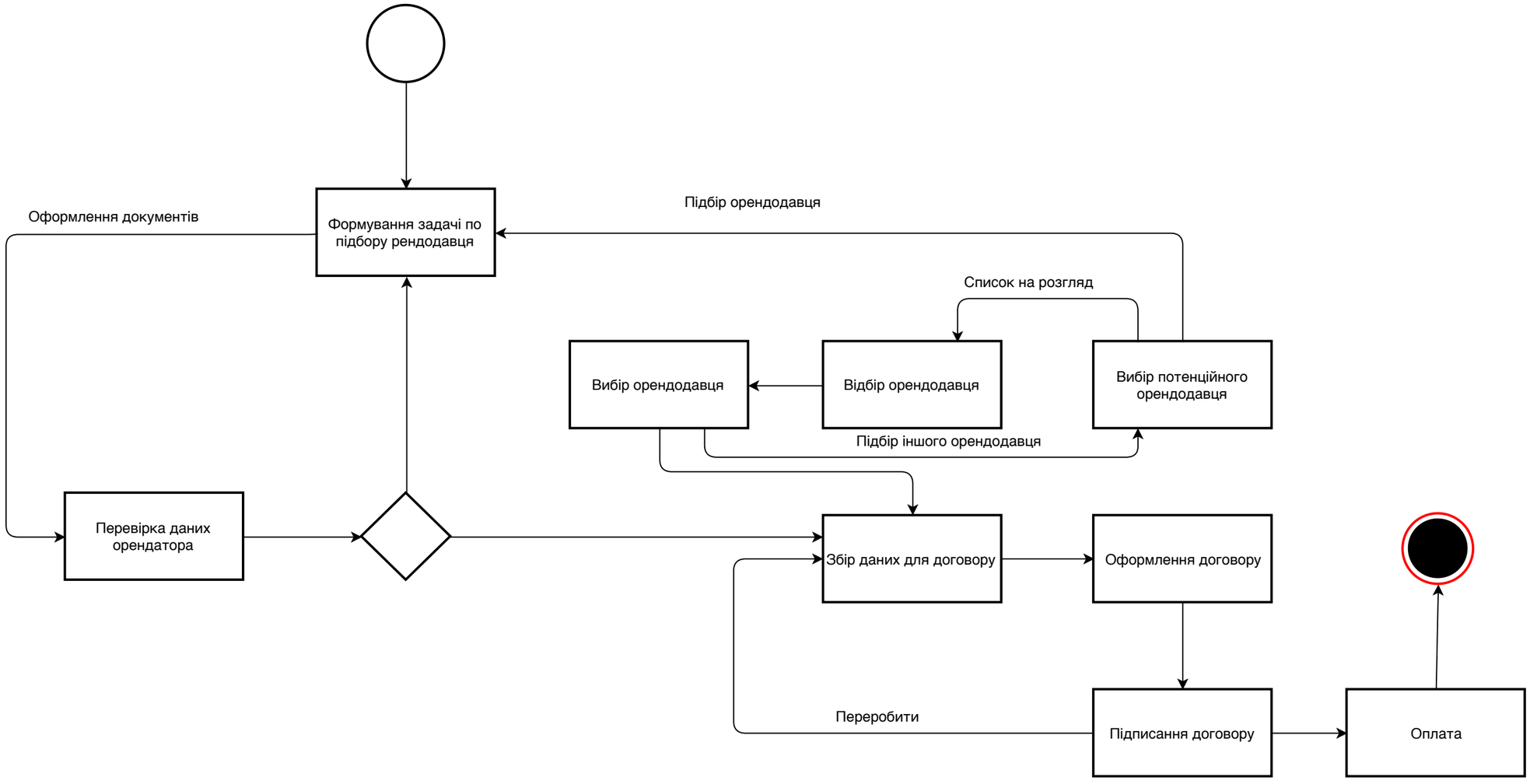
Демонстраційний плакат до магістерської дисертації

«*Інтелектуальна система підтримки вибору житла для покупців та орендаторів*»

Магістрант Меліков Євгеній Олексійович

Керівник Тєлишева Тама Олексіївна

Діаграма станів



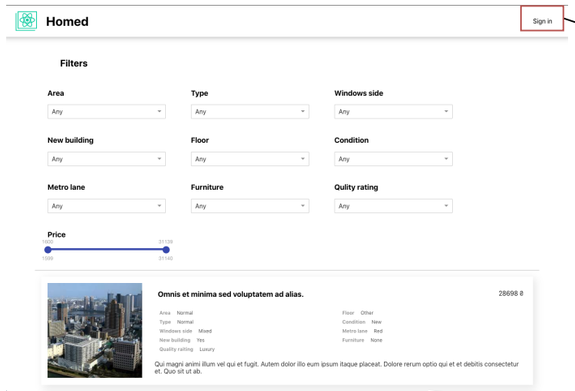
Демонстраційний плакат до магістерської дисертації

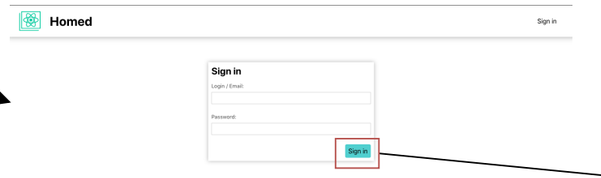
«*Інтелектуальна система підтримки вибору житла для покупців та орендаторів*»

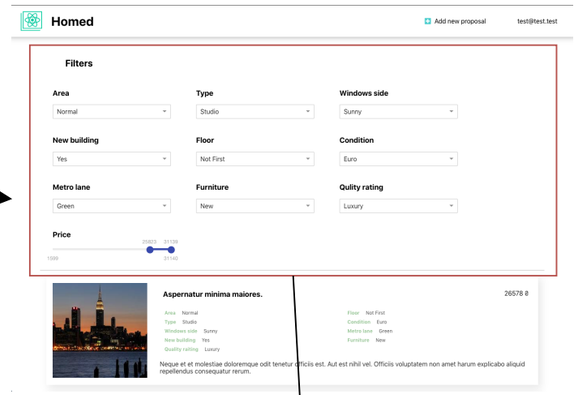
Магістрант Меліков Євгеній Олексійович

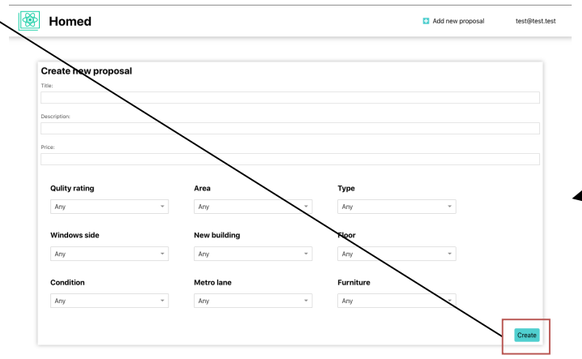
Керівник Тєлишева Тама Олексіївна

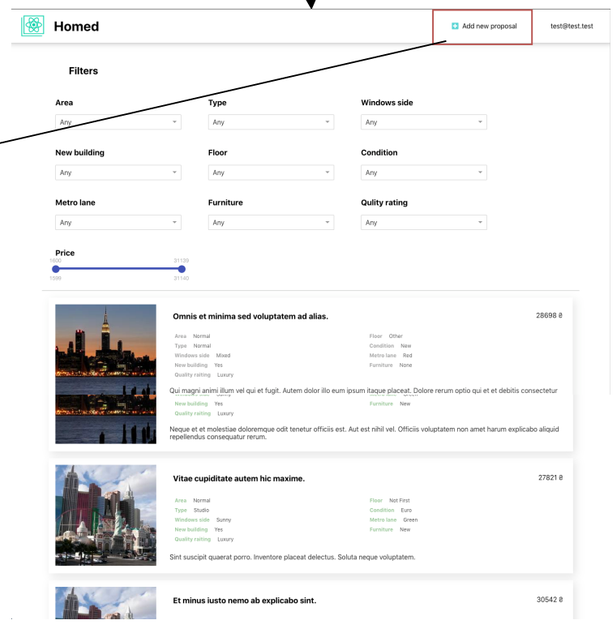
Єкранні форми











Демонстраційний плакат до магістерської дисертації

«*Інтелектуальна система підтримки вибору житла для покупців та орендаторів*»

Магістрант Меліков Євгеній Олексійович

Керівник Тєлишева Тама Олексіївна